

STRUCTURE DU CODE DE PROGRAMMATION

THÈSE

présentée

à la FACULTÉ DES SCIENCES de l'UNIVERSITÉ de NANCY
pour l'obtention
du DOCTORAT de SPÉCIALITÉ (Mathématiques 3^{me} cycle)

par

Marion CRÉHANGE

et soutenue en mars 1961 devant la Commission d'examen

Jury : MM. J. LEGRAS, *Président.*

M. HERVÉ
J. GOSSE } *Examineurs.*

UNIVERSITE DE NANCY

FACULTE DES SCIENCES

STRUCTURE

DU CODE DE PROGRAMMATION

par

Marion CRÉHANGE



UNIVERSITE DE NANCY --- FACULTE DES SCIENCES

Doyen : M. URION
Assesseur : M. ECHEVIN

Doyens honoraires : MM. CORNUBERT, DELSARTE

Professeurs honoraires : MM. GUTTON, CROZE, RAYBAUD, LAFFITTE, LERAY, JOLY, LAFORTE, EICHHORN, CAPELLE, GODEMENT, DUBREIL, L. SCHWARTZ, DIEUDONNE, de MALLEMANN, LONGCHAMBON, LETORT, DODE, GAUTHIER, GOUDET, OLMER, CORNUBERT, CHAPELLE, GUERIN.

Maîtres de conférences honoraires : MM. RAUX, LIENHART.

PROFESSEURS

MM.			
URION	Chimie biologique	DUVAL	Chimie
DELSARTE	Analyse supérieure	COPPENS	Radiogéologie
ROUBAULT	Géologie	FRUHLING	Physique
VEILLET	Biologie animale	LIONS	M. M. P.
ECHEVIN	Botanique	SUHNER	Physique expérimentale
WAILL	Chimie org. industr.	HILLY	Géologie
BARRIOL	Chimie théorique	LE GOFF	Génie Chimique
BIZETTE	Physique	CHAPON	Chimie biologique
GUILLIEN	Electronique	HEROLD	Chimie industrielle
GIBERT	Chimie physique	SCHWARTZ	Exploitation minière
HERVE	Calcul diff. et intégral	GAYET	Physiologie
LEGRAS	Mécanique rationnelle	MANGENOT	Phytopathologie
DAVID	Chimie organique	MALAPRADE	Chimie
BOLFA	Minéralogie	HADNI	Physique
NICLAUSE	Chimie	DELAMAIE -	Zoologie
FAIVRE	Physique appliquée	DEBOUTEVILLE	
AUBRY	Chimie minérale	BONVALET	Mécanique physique

MAÎTRES DE CONFERENCES

MM.			
WERNER	Botanique	RENARD	Physique théor. et Nucl.
GARNIER	Agronomie	CONDE	Zoologie
BRUHAT	Mathématiques génér.	GOSSE	Génie chimique
GUDEFIN	Physique	CHAMPIER	Physique
Mme ROIZEN	Physique	GAY	Chimie biologique
REGNIER	Physico-chimie	CLIN	Paléontologie
KERN	Minéralogie	ROCCI	Géologie
WEPPE	Minéralogie appliquée	WEISS	Physique M. P. C.
BERNARD	Géologie	Mme BASTICK	Chimie M. P. C.
ARAGNOL	Mathématiques (prop.)	VUILLAUME	Psychophysiologie
BONVALET	Mécanique Appliquée	PIERRET	Maître de conf. adjoint
BASTICK	Chimie	PLAN	Mathématiques
MARI	Chimie (ISIN)	LAFON	Physique (ISIN)

Secrétaire : M. CARON

Je tiens tout d'abord à exprimer mes plus vifs remerciements à Monsieur le Professeur LEGRAS pour la façon attentive et bienveillante avec laquelle il a dirigé mon travail et pour les conseils et suggestions qu'il n'a cessé de me prodiguer.

Je remercie également Messieurs les Professeurs M. HERVE et J. GOSSE qui ont bien voulu me faire l'honneur de composer le Jury.



Digitized by the Internet Archive
in 2026

https://archive.org/details/IA41547403_0011

TABLE des MATIERES

I - <u>INTRODUCTION</u>	1
II - <u>LOGIQUE EXTERIEURE ET AUTOPROGRAMMATION</u>	3
II - A - Utilité d'un langage autre que le langage machine	3
II - B - Différence entre logique extérieure et autoprogrammation	4
II - C - Avantages et inconvénients de la logique extérieure et de l'autoprogrammation	5
II - D - Avantages de la coexistence des 2 procédés	6
III - <u>ORGANISATION GENERALE DU C.D.P.</u>	8
III - A - <u>Structure</u>	8
III - A - 1 - Partie centrale	8
III - A - 2 - Sous-programmes - Divers C.D.P.	9
III - B - <u>Paquets de cartes</u>	10
IV - <u>INSTRUCTIONS DU C.D.P. (tronc commun)</u>	11
IV - A - <u>Structure des ordres</u>	11
IV - A - 1 - Ordres à 2 mémoires	11
IV - A - 2 - Ordres à 1 mémoire	12
IV - B - <u>Détail des ordres</u>	13
IV - B - 1 - Ordres à 2 mémoires	13
IV - B - 2 - Ordres à 1 mémoire (ordres de service)	16
V - <u>ECRITURE DES PARTIES INTERPRETATIVES</u>	18

.../...

VI - PARTIE INTERPRETATIVE EN LOGIQUE EXTERIEURE :

GENERALITES

VI - A - <u>Schéma général</u>	20
VI - B - <u>Analyse</u>	21
VI - C - <u>Organigramme</u>	22
VI - C - 1 - Instructions à 1 mémoire	22
VI - C - 2 - Instructions à 2 mémoires	24
a) Partie commune	25
α) Découpage des ordres	25
β) Indexage	25
γ) Partie commençant à CØMMU = 1739	26
δ) Aiguillage "aller à GOOOP"	26
b) Branche particulière à chaque valeur de P	27
α) P = 0	27
β) P = 1 à 5	28
γ) P = 8	29
δ) P = 9	29
c) Analyse	30

VII - ETUDE DETAILLEE DE LA PARTIE INTERPRETATIVE

VII - A - <u>Partie commune à tous les ordres</u>	31
VII - B - <u>Ordres à 1 mémoire ou ordres de service</u>	31
VII - B - 1 - Partie commune à ces ordres	31
VII - B - 2 - Partie propre à chaque code PP'	32
VII - B - 3 - Remarque sur les ordres de service	37
VII - C - <u>Ordres à 2 mémoires</u>	38
VII - C - 1 - <u>Partie commune à tous ces ordres</u>	38
a) Début	38
b) Index et fin de la partie commune	40
c) Analyse	43

VII - C - Ordres à 2 mémoires (suite)

VII - C - 2 - P = 0 : sous-programmes à 2 adresses. Opérations en virgule fixe - Tests 45

- a) PP' = 00 : sous-programmes à 2 adresses
(standardisation et programme interprétatif relatif à ces ordres) 45
- b) PP' = 01 : addition en virgule fixe 48
- c) PP' = 02 : soustraction en virgule fixe 49
- d) PP' = 03 et PP' = 05 : multiplication sans ou avec
décalage 50
- e) PP' = 04 et PP' = 06 : division sans ou avec
décalage 53
- f) PP' = 07 : test de signe à 3 branches 56
- g) PP' = 08 : test sur l'existence d'un 8 ou d'un 9
en position n° B de la mémoire C 60

VII - C - 3 - P = 1, 2, 3, 4, 5 : sous programmes à 3 adresses 61

- Standardisation
- Programme interprétatif relatif à ces ordres

VII - C - 4 - P = 9 : bouclage 63

VII - D - Sous-programmes placés à poste fixe 73

VII - D - 1 - Perforation (49 N iA jB kC) 73

VII - D - 2 - Modification de bouclage (00 N 01937 jB kC) 79

VIII - ASSEMBLAGE ET DISPOSITION DE LA LOGIQUE EXTERIEURE 81

IX -	<u>AUTOPERFORATION - GENERALITES</u>	85
IX - A -	<u>Emplacement du programme en langage machine obtenu.</u> <u>Séquence l_{min}, l_{max}</u>	86
IX - B -	<u>Notations</u>	87
IX - C -	<u>Préparation des cartes à autoperforer</u>	87
IX - D -	<u>Schéma général</u>	88
IX - E -	<u>Organigramme</u>	89 et 89 bis
IX - E - 1 -	Perforation	89
IX - E - 2 -	Instructions à 1 mémoire	90
IX - E - 3 -	Instructions à 2 (ou 3) mémoires	91
	a) Partie commune	91
	b) Branche particulière à chaque valeur de P	91
	α) P = 0	91
	β) P = 1 à 5	92
	γ) P = 8	92
	δ) P = 9	92
X -	<u>ETUDE DETAILLEE DE L'AUTOPROGRAMMATION</u>	93
X - A -	<u>Indexage</u>	93
X - A - 1 -	Indexage des opérations en virgule fixe	94
X - A - 2 -	Indexage des entrées dans les sous-programmes à 2 adresses	97
X - A - 3 -	Indexage des entrées dans les sous-programmes à 3 adresses	99
X - B -	<u>Perforation</u>	101
X - C -	<u>Partie commune à tous les ordres</u>	103
X - D -	<u>Ordres à 1 mémoire ou ordres de service</u>	105
X - D - 1 -	Partie commune à ces ordres	105
X - D - 2 -	Partie propre à chaque code PP'	106
X - E -	<u>Ordres à 2 (ou 3) mémoires</u>	118
X - E - 1 -	<u>Partie commune à ces ordres</u>	118
X - E - 2 -	<u>P = 0 : sous-programmes à 2 adresses. Opérations en virgule fixe. Tests.</u>	120

.../...

X - E - Ordres à 2 (ou 3) mémoires (suite)

a) $P' = 0$: sous-programmes à 2 adresses	120
b) Partie commune aux ordres OP' ($P' \neq 0$)	123
c) Opérations en virgule fixe et tests, non indexés	123
α) Programmes en langage machine élaborés par l'autoprogrammation	123
β) Partie commune aux ordres OP' non indexés	126
χ) $P' = 1$: addition en virgule fixe (sans index)	127
δ) $P' = 2$: soustraction en virgule fixe (sans index)	128
ϵ) $P' = 3$: multiplication sans décalage (sans index)	128
η) $P' = 5$: multiplication avec décalage (sans index)	129
θ) $P' = 4$: division sans décalage (sans index)	131
λ) $P' = 6$: division avec décalage (sans index)	131
μ) $P' = 7$: test SI (sans index)	132
ν) $P' = 8$: test SD (sans index)	133
d) Opérations en virgule fixe et tests, indexés	134
α) Programmes en langage machine élaborés par l'autoprogrammation	134
β) Partie commune aux ordres OP' indexés	139
χ) $P' = 1$: addition en virgule fixe (avec index)	140
δ) $P' = 2$: soustraction en virgule fixe (avec index)	140
ϵ) $P' = 3$: multiplication sans décalage (avec index)	141
η) $P' = 5$: multiplication avec décalage (avec index)	142
θ) $P' = 4$: division sans décalage (avec index)	143
λ) $P' = 6$: division avec décalage (avec index)	143
μ) $P' = 7$: test SI (avec index)	145
ν) $P' = 8$: test SD (avec index)	147

.../...

X - E - 3 - <u>P = 1, 2, 3, 4, 5 : sous-programmes à 3 adresses</u>	148
X - E - 4 - <u>P = 9 : bouclage</u>	151
a) Instructions produites par l'autoprogrammation	152
b) Auto-programmation de l'ordre de bouclage	153
c) Sous-programme de bouclage pour l'exécution en langage machine	154
X - F - <u>Sous-programmes placés à poste fixe</u>	160
X - F - 1 - Perforation (ordre 49 N iA jB kC)	160
X - F - 2 - Modification de bouclage (ordre 00 N 01937 jB kC)	163
 XI - <u>DISPOSITION DE L'AUTOPROGRAMMATION ET DES SOUS-PROGRAMMES DES "PAQUETS VERTS"</u>	164
XI - A - <u>Autoprogrammation</u>	164
XI - B - <u>Exécution en langage machine (paquets verts)</u>	165
 XII - <u>SOUS-PROGRAMMES ADAPTES AU C.D.P. - DIVERS C.D.P. FORMÉS.</u>	
XII - A - <u>Généralités</u>	166
XII - B - <u>C.D.P. en virgule flottante simple précision</u>	169
XII - C - <u>C.D.P. en virgule flottante double précision</u>	171
XII - D - <u>C.D.P. en virgule fixe</u>	171
XII - D - 1 - <u>Sous-programme auxiliaire de décalage</u>	172
a) Sous-programme auxiliaire de décalage pour l'exécution en langage machine	175
b) Sous-programme auxiliaire de décalage en logique extérieure.	177
XII - D - 2 - <u>Ordres particuliers au C.D.P. en virgule fixe</u>	180
 XIII - <u>REMARQUES DIVERSES</u>	182
XIII - A - <u>Chargement</u>	182
XIII - B - <u>Formations d'ordres</u>	187
XIII - C - <u>Indexage double</u>	188
XIII - D - <u>Aiguillage multiple</u>	188
 XIV - <u>CONCLUSION</u>	189

CODE DE PROGRAMMATION

=====

I - INTRODUCTION

Le code de programmation ou "C. D. P." est à la fois une logique extérieure et un dispositif d'autoprogrammation ou compilateur. Il est adapté à l'ordinateur I. B. M. 650 standard mais se généralisera probablement aux autres 650.

La logique extérieure et l'autoprogrammation constituaient chacune déjà un gros progrès sur les simples ordres en langage machine. Mais un procédé comportant une juxtaposition des deux méthodes possède de nets avantages supplémentaires. Le C. D. P. a sur les logiques extérieures classiques la supériorité de permettre une exécution bien plus rapide des programmes grâce à l'autoprogrammation. Par rapport aux dispositifs d'autoprogrammation habituels, il offre la commodité de permettre une analyse en langage "extérieur" et non en langage machine, ce qui rend la mise au point aisée et les modifications peu risquées.

Le C. D. P. a pour autre caractéristique d'être constitué par un tronc commun constant (mais susceptible d'améliorations et d'agrandissements) autour duquel on peut grouper toutes sortes de sous-programmes. Ceci permet, par le choix des sous-programmes, de construire très facilement des codes de programmation adaptés à des problèmes

.../...

particuliers. Pour l'instant, 3 C.D.P. ont été créés : le C.D.P. en virgule fixe, le C.D.P. en virgule flottante simple précision, le C.D.P. en virgule flottante double précision. Chacun de ces C.D.P. peut être modifié et augmenté à volonté (dans la limite de capacité du tambour). D'autre part il est en général très aisé de passer d'un programme écrit en C.D.P. virgule fixe à un programme en C.D.P. virgule flottante (et vice versa) ou d'un programme écrit en C.D.P. virgule flottante simple précision à un programme en C.D.P. virgule flottante double précision (et vice versa). Ceci permet de faire des vérifications de la position de la virgule, ou des améliorations de précision.

Enfin le C.D.P. simule 9 registres d'index. Ses instructions, pour la plupart, occupent chacune 2 mémoires consécutives. Il comporte des ordres logiques commodes, en particulier un ordre de bouclage assez puissant qui exécute une modification automatique d'un registre d'index choisi.

II - LOGIQUE EXTERIEURE et AUTOPROGRAMMATION

II - A - UTILITE D'UN LANGAGE AUTRE QUE LE LANGAGE MACHINE :

Les ordres en langage machine sont des ordres élémentaires, peu pratiques pour la programmation à cause surtout de leur manque de concision. Pour commander une addition en virgule fixe, par exemple, il faut 3 ordres élémentaires; pour calculer une fonction, il faut, dans les meilleures conditions, entrer dans un sous-programme, ce qui exige plusieurs ordres en langage machine. La logique extérieure ou l'autoprogrammation permettent de faire interpréter par la machine des ordres plus compacts que les ordres en langage machine. Une addition, par exemple, s'exprimera par un seul ordre, de même que le calcul d'une fonction classique. De plus, dans le langage plus concis que comprend alors la machine, peuvent entrer des ordres logiques commodes. On peut très bien, d'autre part, créer des logiques extérieures ou des compilateurs adaptés à des programmes particuliers (par exemple le FLEC, spécialement adapté aux problèmes électriques : THOMAS, ingénieur E. N. S. E. M docteur de 3ème cycle de Mathématiques Appliquées).

Si par exemple nous faisons la convention que l'ordre complexe 11 A B C représente l'addition en virgule flottante des nombres contenus respectivement en mémoires A et B pour placer le résultat en C, il est facile de concevoir un programme qui interprétera 11 comme "entrée dans le sous-programme d'addition en virgule flottante" et utilisera les valeurs de A, B et C pour fournir à ce sous-programme les renseignements nécessaires. Ainsi à chaque ordre complexe la partie interprétative fait correspondre une suite d'ordres en langage machine. Ce processus peut se dérouler de 2 façons différentes : en logique extérieure et en autoprogrammation.

.../...

II - B - DIFFERENCE ENTRE LOGIQUE EXTERIEURE ET AUTOPROGRAMMATION

En logique extérieure, la machine traduit chaque ordre complexe en instructions "langage machine" qu'elle exécute aussitôt. Elle passe ensuite à l'ordre suivant qu'elle interprète (c'est-à-dire traduit en langage machine) puis exécute. Il ne reste aucune trace des instructions en langage machine formées. En particulier, si par bouclage on revient à un ordre déjà traité, la machine le retraduit, ignorant le travail effectué auparavant.

En autoprogrammation au contraire, la machine traduit chaque ordre complexe par des ordres en langage machine, mais, pour conserver ces ordres, les perfore sur des cartes après leur avoir attribué des adresses. La succession des événements est donc la suivante : interprétation d'un ordre puis perforation ; interprétation de l'ordre suivant et perforation ; etc, A la suite de cette phase (dans laquelle les bouclages ne sont pas exécutés) qui est l'autoprogrammation proprement dite ou l' "autoperforation", on obtient un paquet de cartes portant le programme en langage machine équivalent au programme écrit initialement en ordres complexes.

Remarque : en C. D. P. , la plupart des ordres sont en fait des entrées dans des sous-programmes. L'autoprogrammation ne provoque pas la perforation des sous-programmes eux-mêmes, car cela risquerait de donner un encombrement énorme au programme en langage machine obtenu : certains sous-programmes seraient en effet reperforés un grand nombre de fois. Seules les instructions nécessaires à l'entrée dans ces sous-programmes sont perforées. Pour exécuter le programme en langage machine obtenu il faudra donc lui adjoindre les sous-programmes utiles (cf paragraphe III - B).

... ..

... / ...

Après avoir effectué l'autoprogrammation, il reste, dans une 2ème phase, à exécuter le programme obtenu. Pour cela, il faut charger les cartes qui ont été perforées par la machine, ainsi que, éventuellement, les sous-programmes utiles. On dispose alors d'un programme en langage machine, sans partie interprétative donc d'exécution beaucoup plus rapide qu'un programme en logique extérieure.

II - C - AVANTAGES ET INCONVENIENTS DE LA LOGIQUE EXTERIEURE ET DE L'AUTOPROGRAMMATION

La logique extérieure a le gros avantage de permettre une analyse très commode du programme et par conséquent une mise au point aisée. En effet l'analyse d'un programme en logique extérieure donne lieu à la perforation d'une carte d'analyse par ordre en langage extérieur. L'analyse est donc beaucoup plus concise et par conséquent bien plus facile à lire et à vérifier qu'une analyse en langage machine. En particulier la suppression de l'analyse des sous-programmes internes est automatique. Le fait que la mise au point soit aisée entraîne la facilité et le peu de risque des modifications.

Par contre, la logique extérieure a un gros inconvénient : la lenteur d'exécution. Pour chaque ordre complexe, la machine doit exécuter une partie interprétative, même si cet ordre a déjà été effectué auparavant. En particulier, dans un bouclage, si l'on passe 1000 fois par l'exécution de chaque ordre, on passe 1000 fois par sa partie interprétative, ce qui cause une grande perte de temps (voir le paragraphe II - B).

L'autoprogrammation, elle, se prête mal à la mise au point. En effet l'analyse d'un programme se fait lors de son exécution. Or c'est au cours de la 2ème phase que les programmes s'exécutent. Ils sont alors en langage machine et il serait très délicat de prévoir pour eux une analyse aussi concise que celle en logique extérieure.

Par contre, pour les programmes répétitifs ou d'utilisation fréquente, l'autoprogrammation représente un gain de temps considérable (qui peut aller de 75 % à 25 % environ suivant le genre de programme et l'importance des sous-programmes) par rapport à la logique extérieure : chaque ordre n'est interprété qu'une seule fois, quel que soit le nombre de fois où il doit être exécuté. En effet, ce n'est que dans la 1ère phase, celle de perforation, qu'il y a un programme interprétatif. Or pendant cette phase les bouclages ne fonctionnent pas. Ainsi les ordres qui doivent, par bouclage, être exécutés 1000 fois ne sont interprétés qu'une seule fois. C'est pendant la 2ème phase, celle où le programme est en langage machine et ne nécessite plus de partie interprétative, que les bouclages fonctionnent. Un autre gain de temps est dû au fait qu'un programme une fois autoprogrammé peut être réutilisé à volonté, toute phase interprétative étant supprimée ; ceci n'est absolument pas le cas en logique extérieure où il faut tout recommencer à chaque utilisation du programme.

II - D - AVANTAGES DE LA COEXISTENCE DES 2 PROCÉDÉS.

Le C. D. P. est l'association d'une logique extérieure et d'un procédé d'autoprogrammation (compilateur). Les ordres complexes propres au C. D. P. peuvent en effet être interprétés soit en logique extérieure soit en autoprogrammation. Ceci permet de bénéficier pour un même programme de la commodité de mise au point en logique extérieure et de la rapidité

d'exécution après l'autoprogrammation. La plupart des programmes seront traités d'abord en logique extérieure pour être vérifiés puis, s'ils sont répétitifs ou d'utilisation fréquente, seront transformés par l'autoprogrammation en programmes en langage machine.

=====

.../...

.....

.....

III - ORGANISATION GENERALE DU C. D. P.

III - A - STRUCTURE

Dans ce chapitre, on entend par C. D. P. aussi bien le C. D. P. "logique extérieure" que le C. D. P. "autoprogrammation".

Le C. D. P. est constitué d'une part d'une partie centrale (tronc commun) comportant les 4 opérations en virgule fixe et les ordres logiques ou de routine, et d'autre part de sous-programmes à 2 ou 3 adresses. Cette structure permet une très grande souplesse du code de programmation et une possibilité énorme d'extension, limitée seulement par la capacité du tambour.

III - A - 1 - Partie centrale

La partie centrale, qui sera étudiée en détail ultérieurement, comporte principalement :

- les opérations en virgule fixe : addition, soustraction, multiplication avec ou sans décalage, division avec ou sans décalage. Ces ordres sont indexables.
- des ordres logiques et de routine : arrêt, saut inconditionnel, retour en langage machine, lecture, perforation, tests (dont un test de signe à 3 branches), ordre de bouclage puissant pouvant modifier automatiquement un registre d'index, etc
- des ordres d'entrée dans des sous-programmes. Ces ordres sont indexables.

.../...

III - A - 2 - Sous-programmes - Divers C. D. P.

Pour obtenir un C. D. P. utilisable, il faut ajouter au tronc commun les sous-programmes utiles. Ce sont par exemple des sous-programmes de calcul de fonctions, ou d'opérations en virgule flottante simple ou double précision, etc Mais il serait peu commode d'être obligé de s'occuper de la mise en place des sous-programmes courants chaque fois qu'ils sont utilisés. On a donc constitué divers groupes de sous-programmes pour former, en les ajoutant au tronc commun, des C. D. P. d'utilisation facile. Il en existe trois actuellement (voir le chapitre XII) :

- le C. D. P. en virgule fixe simple précision groupant autour du tronc commun les sous-programmes de calcul en virgule fixe de fonctions courantes : racine carrée, sinus, cosinus, logarithmes, exponentielle, Arc sinus. Ces calculs ont lieu avec ou sans décalage.
- le C. D. P. en virgule flottante simple précision groupant autour du tronc commun les sous-programmes d'opérations et de calcul de fonctions courantes en virgule flottante FLAIR (dans cette convention, 1 est noté 5110000000).
- le C. D. P. en virgule flottante double précision constitué par le C. D. P. en virgule flottante simple précision augmenté d'ordres d'opérations en virgule flottante double précision (2,18) : dans cette convention, 1 est noté 51100000000000000000.

Chacun de ces C. D. P. est extensible à volonté dans la limite de capacité du tambour. D'autre part si un des sous-programmes n'est pas utilisé dans un programme, sa place peut être récupérée.

III - B - PAQUETS DE CARTES

Quand un programme a été écrit en C. D. P. , il est utilisable en logique extérieure et en autoprogrammation. Si l'on veut l'exécuter en logique extérieure , on le charge avec le paquet interprétatif "logique extérieure". Si l'on veut l'interpréter en autoprogrammation , on le charge avec le paquet "autoprogrammation". C'est le choix du paquet qui fixe le choix du procédé.

Comme on l'a vu dans la 2ème partie du chapitre II - B , pour exécuter un programme perforé par l'autoprogrammation , il faut lui ajouter les sous-programmes utiles. En instituant 3 C. D. P. (virgule fixe , virgule flottante simple et double précision) on a établi 3 paquets de sous-programmes à joindre aux programmes autoperforés. Ces paquets sont appelés paquets pour "l'exécution en langage machine" ou "paquets verts".

En résumé , pour chaque modèle de C. D. P. (virgule fixe , virgule flottante , etc. . .) existent 3 paquets de cartes :

- le paquet "logique extérieure" comportant la partie interprétative en logique extérieure et les sous-programmes,
- le paquet "autoprogrammation" ou "autoperforation" comportant la partie interprétative en autoprogrammation,
- le paquet "exécution en langage machine" comportant les sous-programmes.

.../...

IV INSTRUCTIONS DU C. D. P. (tronc commun)

IV - A - STRUCTURE DES ORDRES

La plupart des logiques extérieures adaptées à l'I. B. M. 650 utilisent, comme le langage machine, des ordres à 10 chiffres. Mais ce format a été jugé insuffisant, les ordres étant alors trop pauvres en informations. En particulier les ordres ne peuvent avoir 3 adresses et une variété suffisante de codes, et l'indexage est peu pratique. Si 10 chiffres sont jugés insuffisants, il est à peu près obligatoire de passer à 20 chiffres, ce qui a été fait pour la plupart des ordres du C. D. P. Ces ordres occupent 2 mémoires consécutives. Cependant certains ordres dits "ordres de service" ne nécessitent pas l'encombrement de 2 mémoires : ces ordres n'ont que 10 chiffres. Enfin, un petit nombre d'ordres à 2 mémoires nécessitent des indications supplémentaires (décalages). On place alors ces indications dans une 3ème mémoire qui suit les 2 autres.

IV - A - 1 - Ordres à 2 mémoires

Ces ordres sont des ordres à 3 adresses, chaque adresse étant indexable. Le C. D. P. simule 9 registres d'index numérotés de 1 à 9. Les registres CTR1, CTR2, ..., CTR9 sont respectivement les mémoires 1968, 1969, ..., 1976. Leur contenu est toujours positif ou nul et inférieur à 1000. Il est de la forme 00 0000 0xxx. Un 10ème registre d'index, CTR0, contient toujours zéro. Il est simulé par la mémoire 1967. Après le chargement du C. D. P., les registres d'index ne sont pas à zéro. Ils le seront par l'ordre d'initialisation (voir plus loin). Seul le CTR0, c'est-à-dire la mémoire 1967, est dès le début à zéro.

Lorsque nous considérerons un ordre quelconque en C. D. P. , nous supposerons que son adresse est n . Si c'est un ordre à 2 mémoires, il sera contenu en n et $n + 1$. La composition des ordres à 2 mémoires est la suivante :

PP' N i A	j B k C
x x xxx x xxxx	x xxxx x xxxx
Positif	Signe ignoré
Contenu en n	Contenu en $n + 1$

- PP' est un code d'opération
- N est l'adresse de l'instruction suivante (comprise entre 000 et 999)
- A, B et C sont des adresses (indexables)
- i, j, k, compris entre 0 et 9, indiquent respectivement qu'il faut ajouter à l'adresse A le contenu du registre d'index n° i, à B celui du n° j, à C celui du n° k.

Remarque : les ordres indexés sont exécutés en tenant compte des index mais ne sont pas modifiés eux-mêmes (ils restent intacts dans leurs mémoires).

IV - A - 2 - Ordres à 1 mémoire

Ils sont négatifs et de la forme

- PP' U V
x x xxxx xxxx

- PP' est un code pouvant varier entre 00 et 15. Pour l'instant, seuls les codes 00 à 09 ont été utilisés (sauf 08).
- U et V sont des adresses non indexables, assimilables dans la plupart des cas respectivement à l'adresse facteur et à l'adresse instruction suivante d'un ordre en langage machine.

.../...

IV - B - DETAIL DES ORDRES

IV - B - 1 - Ordres à 2 mémoires

00 N 0A jB kC Entrée dans un sous-programme en langage machine à 2 adresses (calcul de fonction).

A est l'adresse de la 1ère instruction du sous-programme. La donnée se trouve en B ; le résultat est envoyé en C (au sujet de la standardisation de ces sous-programmes, voir le paragraphe VII - C - 2 - a).

Remarque : le sous-programme peut très bien être conçu pour utiliser des données placées en B, B + 1, , et envoyer des résultats en C, C + 1,

<u>01 N iA jB kC</u>	(A) + (B) en C	} en virgule fixe
<u>02 N iA jB kC</u>	(A) - (B) en C	

Pour ces 2 ordres, si le résultat a plus de 10 chiffres, il y a "dépassement de capacité" comme en langage machine.

03 N iA jB kC (A) x (B) en C

- Si le produit comporte plus de 10 chiffres (la machine fait un test GNN), la machine s'arrête avec au pupitre 01 1333 xxxx. En logique extérieure, on peut repartir en appuyant sur "départ programme". Après ce départ, le contenu de l'accumulateur droit (produit incomplet) est transféré dans la mémoire C et le programme continue. L'ordre est analysé normalement.
- Il est commode de pouvoir commander des décalages sur le produit avant le test. Dans ce cas on utilise l'ordre 05 au lieu de 03.

04 N iA jB kC (A) : (B) en C

- Pour effectuer la division, le programme met le dividende (A) dans l'accumulateur droit.

.../...

. Si le quotient est trop grand, il y a un "dépassement de capacité" comme en langage machine.

. Il est commode de pouvoir commander des décalages sur le dividende. Dans ce cas, on utilise l'ordre 06 au lieu de 04.

05 N iA jB kC (A) x (B) en C avec décalage. Ordre à 3 mémoires.

- L'ordre de décalage à exécuter sur le produit doit être placé dans la mémoire qui suit celle contenant jB kC. Il doit avoir la forme suivante :

3 x 000 m 1939 ; cet ordre, à part l'adresse instruction suivante, est exactement l'ordre de décalage en langage machine qui s'exécute (x = 0 ou 1 ou 5 ou éventuellement 6).

- L'arrêt 01 1333 xxxx peut aussi se produire pour l'ordre 05.

06 N iA jB kC (A) : (B) en C avec décalage. Ordre à 3 mémoires.

L'ordre de décalage (en langage machine) doit être placé dans la mémoire qui suit celle contenant jB kC, et doit avoir la forme suivante : 3x 000m 1991. Il s'exécute avant la division.

07 N OA OB kC Aiguillage à 3 branches (test SI)

Si (C) < 0 Aller à N

Si (C) = 0 aller à A

Si (C) > 0 aller à B

L'ordre d'exécution des tests est : 1° ANG ; 2° GNN.

08 N OA OB kC Saut conditionnel suivant une position de la mémoire C.

B a la forme suivante : 000B. La machine amène le contenu de C dans le distributeur et fait un test SD sur la position n° B de ce distributeur. B = 0 désigne la position 10. Si c'est un 8, le programme est aiguillé en N ; si c'est un 9, il va en A ; si ce n'est ni un 8 ni un 9, le programme s'arrête sur le test SD. L'adresse C peut être 8000.

.../...

PP' N iA jB kC $P = 1, 2, 3, 4, 5 - PP' \neq 49$.

Entrée dans un sous-programme à 3 adresses.

Les données se trouvent en A et B, le résultat est envoyé en C. L'adresse de la 1ère instruction du sous-programme doit être placée, pour la logique extérieure et l'autoprogrammation, en $1875 + PP'$ sous la forme 00 0000 xxxx. (Pour la standardisation, voir le paragraphe VII - C - 3).

Remarque : le sous-programme peut être conçu pour utiliser des données placées en A, $A + 1, \dots$ et B, $B + 1, \dots$, et envoyer les résultats en C, $C + 1, \dots$.

9i N α H 0B 0C Bouclage

Cet ordre commande : exécuter H fois ($H \gg 1$) le programme commençant en B, en ajoutant chaque fois C (et non son contenu) au contenu du registre d'index $n^\circ i$ et en utilisant le compteur de boucles $n^\circ \alpha$. A la sortie de boucle, aller à N. La boucle a alors été décrite H fois (répétée H - 1 fois). A la sortie de boucle, le compteur de boucles $n^\circ \alpha$ et le registre d'index $n^\circ i$ sont remis à zéro.

Grâce à l'existence de 10 compteurs de boucles, on peut exécuter jusqu'à 10 boucles intérieures les unes aux autres. Des boucles intérieures les unes aux autres doivent avoir des α différents. Si une boucle possède une boucle intérieure, il ne faut pas lui attribuer $\alpha = 0$.

Remarque 1 : si l'on sort de la boucle avant l'exécution des H tours (ceci peut être causé par un test ou un aiguillage à distance), il faut remettre à zéro le compteur de boucles $n^\circ \alpha$ et le registre d'index $n^\circ i$ (si l'on doit ré-utiliser ceux-ci sans refaire d'initialisation). De plus, dans ce cas, il ne faut pas utiliser $\alpha = 0$.

Remarque 2 : si une boucle ne possède pas de boucle intérieure, il est recommandé de lui attribuer $\alpha = 0$ (car alors les calculs sont plus courts pour l'exécution en langage machine) sauf dans le cas cité en remarque 1.

.../...

Remarque 3 : si $i = 0$, aucun registre d'index n'est modifié, ce qui évite de détruire le registre d'index n° 0 qui est par définition à zéro : l'adresse C est alors ignorée.

Remarque 4 : si, par suite d'une erreur, le compteur de boucles n° α n'a pas été remis à zéro avant l'ordre de bouclage et contient une valeur supérieure ou égale à H, la boucle n'est pas répétée : le compteur de boucles et le registre d'index sont remis à zéro et le programme continue en N.

00 N, 01937 jB kC Modification d'un ordre de bouclage.

Cet ordre est un sous-programme à 2 adresses placé à poste fixe. Il remplace le nombre H de boucles de l'ordre de bouclage placé en C et C + 1 par H' contenu en B sous la forme 00 0000 H'.

49 N iA jB kC Perforation d'une séquence en n mots par carte. Cet ordre, qui est un sous-programme à 3 adresses placé à poste fixe, commande la perforation à A mots par carte du contenu des mémoires B à C comprise. Il existe un autre ordre de perforation, - 06 U V.

IV - B - 2 - Ordres à 1 mémoire (ordres de service)

- 00 U V Saut inconditionnel à V.
- 01 U V Arrêt inconditionnel repéré, c'est-à-dire que U apparaît dans le registre programme (01 U xxxx). On peut repartir à V.
- 02 U V Initialisation. Cet ordre provoque la remise à zéro des compteurs de boucles et registres d'index (ainsi que les mémoires 1950 à 1966). En exécution en langage machine, il restaure aussi le contenu de la mémoire Δ bouclage. U est ignoré.
- 03 U V Remise à zéro de la mémoire U.

.../...

- 04 U V Entrée dans le distributeur du contenu de U et arrêt par l'ordre 01 U xxxx. Cet ordre sert à vérifier un résultat en cours de calcul sans faire d'arrêt prédéterminé.
- 05 U V Sortie de code. Cet ordre envoie en langage machine à l'instruction U. En autoprogrammation, la machine perfore le programme en langage machine (voir le paragraphe X - D - 2). Pour rentrer en code, il faut affecter de l'adresse instruction 1800 le dernier ordre en langage machine.
- 06 U V Perforation. Ordre équivalent à 71 U V en langage machine. Il existe aussi 49 N iA jB kC.
- 07 U V Lecture. Ordre équivalent à 70 U V à condition que les cartes lues soient "non chargement".
- 09 U V Fin de séquence (voir le paragraphe VII - B - 2). Cet ordre envoie à U en logique extérieure et exécution en langage machine, mais à V en autoprogrammation.

.../...

V ECRITURE DES PARTIES INTERPRETATIVES

En logique extérieure et en autoprogrammation, la partie interprétative a été écrite en PASØ II, qui se prête beaucoup mieux aux transformations de programmes et s'écrit plus facilement que le vrai langage machine. De nombreuses modifications qui auraient été pénibles et dangereuses en langage machine se sont faites très aisément en PASØ. Dans la suite, chaque adresse sera désignée par son nom en PASØ, suivi de l'adresse réelle attribuée par l'assemblage. Le numéro de l'instruction dans l'assemblage PASØ permet de repérer plus rapidement les instructions sur la liste située à la fin de cette brochure. Ce numéro, précédé de "n°", sera quelquefois indiqué à la suite des adresses.

Certaines mémoires ont la même utilisation en logique extérieure et en "exécution en langage machine" ou en autoprogrammation. Les principales de ces mémoires sont :

- DEBUT = 1800, adresse de la 1ère instruction de la partie interprétative en logique extérieure comme en autoprogrammation.
- MEMNN = 1884, mémoire contenant l'adresse de l'instruction suivante 00 0000 N.
- huit mémoires de travail, W 0000 = 1990 à W 0007 = 1997, employées pour placer des résultats ou des indications intermédiaires. Ces mémoires de travail sont utilisables dans les sous-programmes mais, d'un ordre en C. D. P. à un autre, ne peuvent pas conserver d'indication. Les mémoires W 0008 = 1998 et W 0009 = 1999, non utilisées par les parties interprétatives, mais disponibles, peuvent au contraire conserver des indications d'un ordre à l'autre. Cependant, si l'on exécute une analyse, on ne peut pas utiliser 1999 comme adresse C car alors l'analyse doit chercher aussi le contenu de C + 1 = 2000 et cela provoque une "erreur adresse".

.../...

- les mémoires P 0001 = 1977 à P 0010 = 1986 qui sont des mémoires de perforation utilisées aussi bien par la logique extérieure pour l'analyse que par l'autoperforation.
- les compteurs de boucles H 0000 = 1940 à H 0009 = 1949 et les registres d'index C 0000 = 1967 à C 0009 = 1976.

Les programmes ont été écrits et surtout vérifiés morceau par morceau.

.../...

VI - PARTIE INTERPRETATIVE EN LOGIQUE EXTERIEURE :

GENERALITES

Interpréter un ordre en logique extérieure consiste à former, en tenant compte des indications données par cet ordre, des instructions en langage machine permettant d'effectuer les opérations demandées par l'ordre. Après les avoir exécutées, la machine va chercher l'ordre suivant sur lequel elle recommence ce processus.

VI - A - SCHEMA GENERAL.

L'interprétation d'un ordre en C.D.P. commence toujours par l'instruction placée en DEBUT = 1800 (n° 41). La seule indication demandée par la machine à ce moment est l'adresse n de l'ordre à interpréter. Cette adresse doit être placée sous la forme 00 0000 n dans la mémoire MEMNN = 1884. L'exécution en logique extérieure d'un ordre se passe donc suivant le schéma suivant :

- recherche de l'ordre dans sa ou ses mémoires (commence en 1800)
- découpage de l'ordre et séparation des diverses indications.
- mise en place de l'adresse de l'instruction suivante en MEMNN = 1884
- formation des ordres en langage machine et exécution
- retour à DEBUT = 1800.

La succession de ces phases n'est pas toujours exactement celle-ci.

.../...

VI - D - ANALYSE.

Pour tous les ordres, si le signe du pupitre est -, la machine perfore une carte d'analyse. Cette carte porte les renseignements suivants :

- Pour les ordres à 1 mémoire :

Mot 1 : 00 n 0000

Mot 2 : instruction (en n)

Pour ces ordres, la carte d'analyse sert à signaler la présence de l'ordre dont la fonction est en général évidente.

- Pour les ordres à 2 mémoires sauf le bouclage :

Mot 1 : 00 n 0000

Mot 2 : }
Mot 3 : } instruction (en n et n + 1)

Mot 4 : { Si non indexage : n'importe quoi
 { Si indexage 0 xxx xxx xxx
 (CTRi)(CTRj)(CTRk)

(CTRi) représente le contenu du registre d'index n°i.
Ce mot permet de connaître le contenu de tous les registres d'index. Ce contenu doit toujours être positif et inférieur à 1000.

Mot 5 : Contenu de A

Mot 6 : Contenu de B

Mot 7 : Contenu de C
Mot 8 : Contenu de C + 1 } Résultat

Les mots 5, 6, 7, 8 représentent le contenu des mémoires après l'exécution de l'ordre.

.../...

- Pour le bouclage

Mot 1 : 00 n 0000

Mot 2 : }
Mot 3 : } instruction (en n et n + 1)

Mot 4 : Contenu du compteur de boucles n° α après sa modification.

Mot 5 : Contenu du registre d'index n° i après sa modification.

La perforation de la carte d'analyse se fait dans la bande 1950. Autrement dit, le mot 1 vient de 1977 que nous appellerons P 0001, le mot 2 de 1978 ou P 0002,, le mot 8 de 1984 ou P 0008.

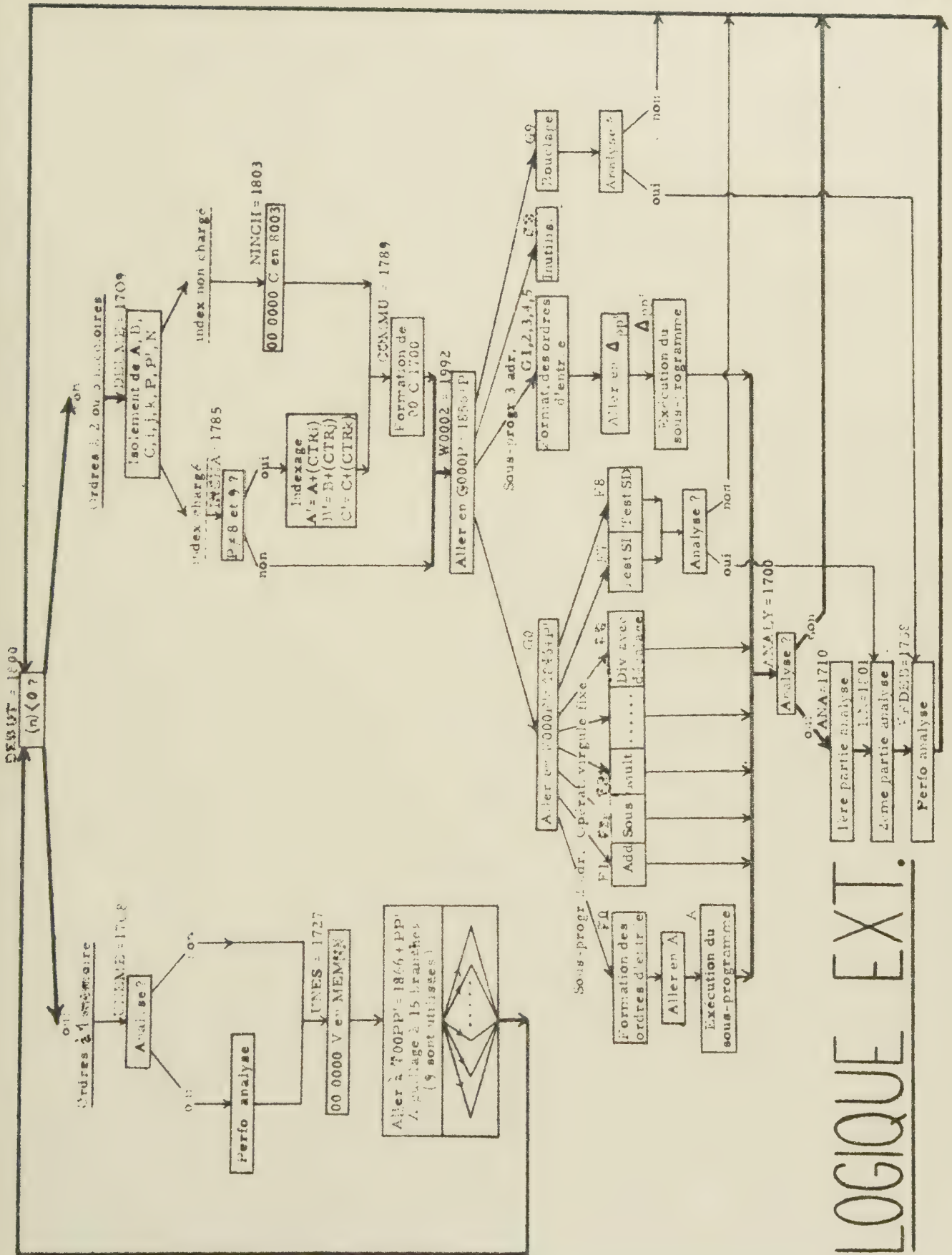
VI - C - ORGANIGRAMME (voir figure page 22bis)

Nous noterons n l'adresse de l'instruction considérée (contenue en n ; ou n et n + 1 ; ou n, n + 1 et n + 2) et N l'adresse de l'instruction suivante. La première opération que fait la machine est d'amener dans l'accumulateur droit le contenu (n) de n et de tester son signe. Dès cet endroit se séparent les trajets suivis pour les instructions à 1 mémoire et celles à 2 mémoires. C'est en effet le signe de (n) qui les différencie.

VI - C - 1 - Instructions à 1 mémoire (cas où le contenu de n est négatif).

La 1ère instruction de cette branche est l'instruction UNEME = 1708 (n° 186). Elle commence une partie dans laquelle ont été groupées toutes les opérations valables à la fois pour tous les ordres de service. Cette partie comporte ;

.../...



- D'abord l'analyse : pour les ordres de service, les seules indications intéressantes sont l'adresse de l'instruction et l'instruction elle-même. Or, dès l'exécution de UNEME = 1708 ces indications sont en place, la première en P 0001 = 1977, la seconde en P 0002 = 1978 (voir le détail de la partie interprétative). On peut donc dès le début traiter le problème de l'analyse. Pour cela, le contenu de la mémoire "pupitre" est amené dans l'accumulateur et testé. S'il est négatif, cas où l'on désire l'analyse, la carte d'analyse est perforée. Sinon aucune carte n'est perforée.

- Ces 2 cas se rejoignent à l'instruction UNES = 1727 où débutent l'envoi de 00 0000 V en MEMNN = 1884, c'est-à-dire la mise en place de l'instruction suivante, et la préparation du test à branches multiples différenciant les branches propres à chaque code PP'. En fait, V n'est pas toujours l'instruction suivante. Mais on a intérêt pour gagner des mémoires à placer son envoi dans la partie commune.

- Le test est un test à 15 branches, laissant ainsi la possibilité d'utiliser 15 codes PP' différents (9 seulement ont été utilisés jusqu'à présent). Ce test est très simple. Il en existe de nombreux autres du même type dans les parties interprétatives, que ce soit en logique extérieure ou en auto-perforation. Il consiste à faire débiter la branche relative à PP' = 00 à une instruction d'adresse: "constante + 00" ; celle relative à PP' = 01

.../...

à "constante + 01" ; etc Cette forme peut être notée par des adresses régionales en PASØ. La région des 1ères instructions des branches relatives aux ordres à 1 mémoire sera par exemple désignée par la lettre T. La branche relative au code PP' commencera donc en $T\ 00PP' = 1866 + PP'$. Pour former l'ordre d'aiguillage (aller à T 00PP') il suffit d'ajouter à 00 0000 00PP' la constante 00 0000 T0000 = 00 0000 1866. Le mot ainsi formé représente alors l'ordre d'aiguillage.

- Exécution du test : il suffit pour cela d'exécuter l'ordre formé, c'est-à-dire $00\ 0000\ T\ 00PP' = 00\ 0000\ 1866 + PP'$.

= Parties propres à chaque code PP' : elles seront passées en revue au cours de l'étude détaillée de la partie interprétative.

- Toutes les parties particulières se terminent par un retour à DEBUT = 1800. La seule exception apparente est celle du code 05 (retour en langage machine). Cette exception n'est que fictive, car la fin de son exécution (c'est-à-dire, au point de vue de la logique extérieure, la fin de l'exécution du programme en langage machine) doit être marquée par un retour à 1800, c'est-à-dire à DEBUT (voir étude détaillée).

VI - C - 2 - Instructions à 2 mémoires : (cas où le contenu de n est positif).

La 1ère instruction de cette branche est l'instruction DEUME = 1709
(n° 48)

a) Partie commune :

La partie commune se décompose en 4 parties principales.

α) Le découpage des ordres. Ce découpage sera envisagé dans l'étude détaillée de la partie interprétative.

β) L'indexage. Pour raccourcir l'exécution en logique extérieure, si aucun ordre d'un programme n'est indexé, on peut supprimer une petite portion de la partie interprétative (matérialisée par un paquet d'indicatif spécial, le paquet "index").

- Si le paquet "index" n'est pas chargé, il ne reste, après le découpage des ordres, rien d'essentiel à faire par la partie interprétative commune à tous les ordres. Par commodité, la machine fabrique, en COMMU = 1789 (n° 79) le mot 00 C 1700 très utilisé par la suite. Avant d'aller à COMMU = 1789, la machine doit exécuter une instruction (NINCH = 1803) qui permet d'être à des états équivalents, lorsqu'on arrive à COMMU = 1789, que le paquet "index" soit chargé ou non.

- Si le paquet "index" est chargé, la partie interprétative doit indexer les adresses. Mais l'indexage n'est pas valable pour les ordres tels que $P = 8$ ou 9 . Pour ces ordres, la partie interprétative commune n'a plus rien à faire après le découpage, si bien que pour $P = 8$ ou 9 , le programme va directement à l'aiguillage gouverné par P .

.../...

Pour les ordres tels que $P \neq 8$ et 9 , l'indexage consiste à ajouter à l'adresse A le contenu du registre d'index n° i, à B celui du n° j et à C celui du n° k. Si un, deux ou les trois indices sont nuls, l'indexage a quand même lieu. Mais, pour cela, il est essentiel que le contenu du registre n° 0 soit nul.

Remarque : il serait possible de faire un test sur la nullité simultanée de i, j, k pour ne pas inutilement exécuter les opérations d'indexage. Mais le gain de temps serait assez minime et la perte de place assez sensible. Or, en logique extérieure, puisqu'il existe une autoprogrammation, le temps n'est qu'un facteur secondaire par rapport à l'encombrement. En autoprogrammation, par contre, ce test a lieu.

- 6) Après l'indexage, la branche "index chargé" rejoint la branche "index non chargé" en CØMMU = 1789. Après la formation de 00 C 1700 (c'est-à-dire en PASO : 00 C ANALY), c'est à l'aiguillage multiple gouverné par P que l'on arrive. L'ordre d'aiguillage a été formé dans la mémoire W 0002 = 1992 au cours du découpage des ordres.
- 8) Aiguillage gouverné par P. Il serait trop encombrant d'effectuer, comme pour les ordres de service, un aiguillage ayant autant de branches qu'il existe de valeurs de PP'. On a donc commandé d'abord un aiguillage suivant P permettant de distinguer les différents genres

d'ordres. Puis, pour certaines valeurs de P, un aiguillage suivant P' est nécessaire. L'aiguillage suivant P envoie à l'instruction d'adresse régionale $G\ 000P = 1856 + P$. Ainsi la branche commençant en $G\ 0000 = 1856$ est relative aux sous-programmes à 2 adresses et aux opérations en virgule fixe. Les branches commençant en $G\ 0001 = 1857$, $G\ 0002 = 1858$, , $G\ 0007 = 1863$ se rejoignent et sont relatives aux sous-programmes à 3 adresses. En fait P = 6 et 7 ne sont pas utilisables ; cette restriction a été apportée pour gagner de la place sur le tambour, car le nombre de 50 sous-programmes à 3 adresses utilisables simultanément a été jugé tout-à-fait suffisant. La branche commençant en $G\ 0008 = 1864$ n'existe pas encore. Elle est réservée à d'éventuels ordres de forme spéciale. Quant à la branche commençant en $G\ 0009 = 1865$, elle est relative à l'ordre de bouclage.

Remarque : Il peut paraître curieux qu'un aiguillage à 10 branches contienne 7 branches se rejoignant aussitôt. C'est cependant le procédé de différenciation qui a paru le plus économique et le plus souple. S'il n'avait pas été employé, il aurait fallu avoir recours à des tests.

b) Branche particulière à chaque valeur de P.

α) P = 0 : la machine fait immédiatement un aiguillage multiple sur P'.

.../...

Les branches particulières à chaque P' commencent à l'adresse régionale $F\ 000P' = 1846 + P'$. Les branches commençant aux adresses $F\ 0001 = 1847$ à $F\ 0006 = 1852$, c'est-à-dire correspondant aux codes $PP' = 01$ à 06 , sont relatives aux ordres d'opérations en virgule fixe. Elles rejoignent la plupart des autres branches avant la partie "analyse". Les branches $F\ 0007 = 1853$ et $F\ 0008 = 1854$ correspondant aux ordres de test $PP' = 07$ et 08 ont leur propre début d'analyse. Elles rejoignent la plupart des autres branches en cours d'analyse ou à $DEBUT = 1800$ suivant que l'on désire ou non l'analyse. La branche $P' = 0$, correspondant aux ordres d'entrée dans les sous-programmes à 2 adresses, commence par la formation des ordres d'entrée dans le sous-programme et se poursuit par l'exécution effective du sous-programme. Elle rejoint la majorité des branches à l'analyse.

- β) $P = 1$ à 5 : il s'agit des ordres d'entrée dans les sous-programmes à 3 adresses. La branche commune à tous ces ordres consiste en la formation de l'entrée dans le sous-programme voulu. Elle est suivie de l'exécution de celui-ci, exécution commençant à l'instruction placée en $\Delta_{pp'}$ (cf chapitre VII - C-3) ; cette adresse représente le facteur de translation du sous-programme, si la 1ère instruction est en $\Delta + 0$. L'adresse $\Delta_{pp'}$ ne peut pas, comme pour les sous-programmes à

.../...

2 adresses, figurer dans l'ordre. Elle est indiquée sous la forme 00 0000 $\Delta_{pp'}$, dans la mémoire 1875 + PP'. Il y a ainsi une correspondance facile entre le code et l'adresse de la première instruction du sous-programme. L'exécution du sous-programme terminée, le programme rejoint la plupart des autres branches au début de l'analyse.

8) P = 8 n'est pas encore créé.

8) P = 9 correspond à l'ordre de bouclage. Cet ordre à structure très spéciale ne peut pas s'adapter à l'analyse commune. Il possède donc son propre programme d'analyse et ne rejoint l'ensemble des branches qu'à l'ordre de perforation de l'analyse ou à DEBUT = 1800 selon que l'on désire ou non l'analyse. Le programme de bouclage a été écrit sous forme d'un sous-programme (non standard) translatable, ce qui a permis, au cours des divers assemblages de la partie interprétative, de garder en formation compacte la partie réservée au bouclage. Le facteur de translation de ce sous-programme est placé dans la mémoire 1875 + 61 = 1936 sous la forme 00 0000 $\Delta_{bou} = 00 0000 1596$. Ce mot représente d'ailleurs l'ordre "aller au sous-programme de bouclage".

.../...

- c) Analyse : pour la plupart des ordres, le même programme d'analyse est utilisé. Ce programme, qui sera étudié en détail plus loin, commence par un test sur le signe du pupitre. Si ce signe est +, le programme d'analyse ne doit pas être déroulé et l'on est envoyé en DEBUT = 1800 où commencera l'interprétation de l'ordre suivant. Si ce signe est -, le programme d'analyse se déroule. Il se termine par la perforation d'une carte et le retour à DEBUT = 1800.

VII - ETUDE DETAILLEE DE LA PARTIE INTERPRETATIVE

Les N° placés en tête des tableaux suivants représentent le numéro de chaque ligne dans l'assemblage.

VII - A - PARTIE COMMUNE A TOUS LES ORDRES : commence en DEBUT = 1800
 Au début de cette partie, se trouve dans la mémoire MEMNN = 1884 la quantité 00 0000 n indiquant l'adresse n de l'ordre à interpréter.

N°	Adresse	C. O.	A. F.	A. I.	(8003)	(8002)	(8001)	
41	DEBUT	R AD	M EMNN		0.....0	0.....0n		
42		D AG	0004			00 n 0000		
43		T RD	P 0001				00 n 0000	en P1 (pour analyse)
44		E DI	G G	G I			65 xxxx GH	
45	G G	65	0000	G H				
46	G I	S BF	G G	8001			65 n GH	en GG
	(8001)	R AD	n	G H	0.....0	(n)		
47	G H	A NG	U NEME	D EUME				Test sur (n)

(UNE Mémoire) (DEUX Mémoires)

Cette séquence de programme sert à former l'ordre 65 n GH qui amène le contenu de n dans l'accumulateur, puis à exécuter cet ordre et à faire le test qui envoie : en UNEME = 1708 si l'ordre est à 1 mémoire et en DEUME = 1709 s'il est à 2 mémoires.

VII - B - ORDRES A 1 MEMOIRE OU ORDRES DE SERVICE

VII - B 1) Partie commune à ces ordres : débute à UNEME = 1708 (n° 186)

					0.....0	-PP' U V	-PP' U V	
186	U NEME	T RD	P 0002				-PP' U V	en P2 (pour analyse)
187		R AG	8000					
188		A NG		U NES				Analyse ?
189		P FØ	P 0001	U NES				Perfo si analyse
190	U NES	R AV	P 0002		0.....0	PP' U V		
191		E DI	8003				0.....0	
192		S BI	M EMNN				00 0000 V	en MEMNN = 1884
193		D AG	0002		0....0PP'	U V 00		
194		A AG	C TE4	8003	0..0(TOOPP')	U V 00	0...0T0000	Formation aiguillage
	(8003)	S IØP	0000	T 00PP'				Aiguillage

- La carte d'analyse porte en mot 1 le contenu de P 0001 = 1977, soit 00 n 0000 et en mot 2 le contenu de P 0002 = 1978, soit - PP' U V.
- V, dans la plupart des cas, représente l'adresse de l'instruction suivante. Le programme envoie donc 00 0000 V en MEMNN = 1884 dans la partie commune car cela est plus rentable au point de vue de l'encombrement.
- La mémoire CTE 4 = 1775 contient 00 0000 T0000 = 00 0000 1806.

VII - B - 2 - Partie propre à chaque code PP' :

= Code - 00 U V : sans opération, aller à V.

217 T 0001 S 0P 0 0 0 0 D EBUT

Le seul but de cette partie de programme est d'envoyer à DEBUT = 1800. En effet, lorsqu'on arrive à DEBUT = 1800, c'est l'adresse V qui est dans la mémoire MEMNN = 1884 car elle a été placée par la partie commune aux ordres à 1 mémoire. C'est donc bien l'instruction V qui est exécutée ensuite.

= Code - 01 U V : arrêt ; après l'arrêt, aller éventuellement à V.

199	T 0001	E DI	A RDEB	D ADSB	0-0 T0001	U V 00 01 0000	DEBUT
197	D ADSB	D AD	0 0 0 2	S BFW1	0... 0xx	xx U V	
198	S BFW1	S BF	W 0 0 0 1	8001		01 U	DEBUT
	(8001)	A RT	U	D EBUT			

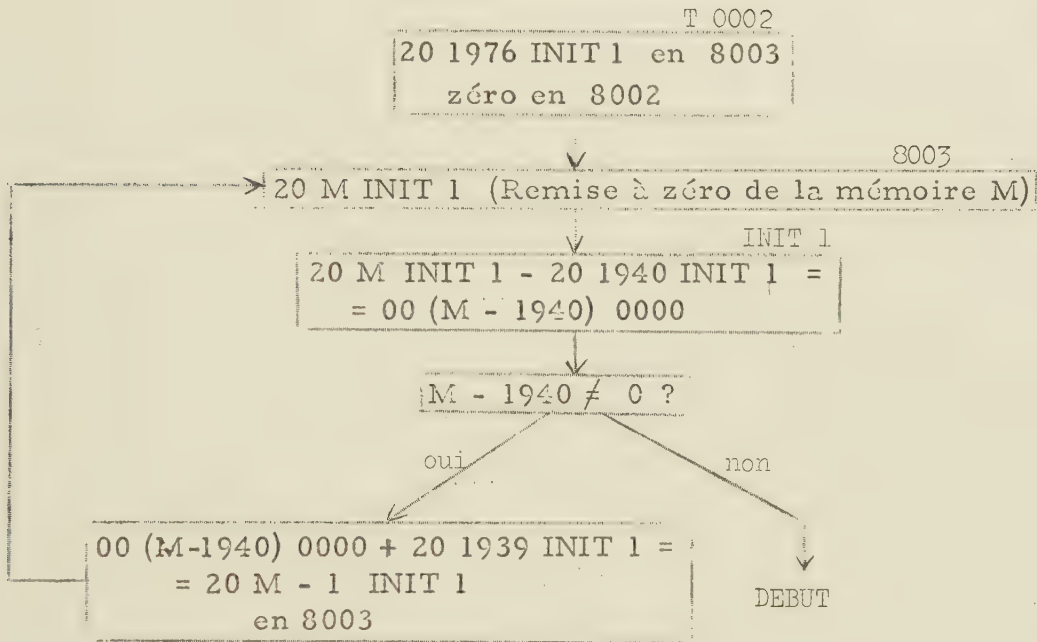
- Les 3 premiers ordres forment l'ordre d'arrêt 01 U DEBUT. Cet ordre a comme adresse facteur U pour pouvoir être repéré. En effet, il est très utile lorsque la machine s'arrête de savoir quel ordre d'arrêt a fonctionné. L'adresse instruction suivante est DEBUT = 1800 pour que, si l'on veut faire repartir le programme, l'instruction V soit exécutée.
- La mémoire ARDEB contient la constante 01 0000 DEBUT = 01 0000 1800.
- Remarque : L'ordre de décalage pourrait s'exécuter avant la mise en place de 01 0000 DEBUT dans le distributeur. Mais cela supprimerait la possibilité d'utiliser l'ordre DADSB pour d'autres codes que - 01 U V.

= Code - 02 U V : Initialisation.

Cet ordre sert à remettre à zéro les mémoires 1940 à 1976, c'est-à-dire les compteurs de boucles et d'index plus les mémoires 1950 à 1966. On verra qu'en exécution en langage machine (après autoprogrammation) cet ordre a un rôle supplémentaire.

.../...

L'organigramme suivant montre le déroulement de la partie interpré-
tative propre à l'ordre - 02 U V -



206	T	0002	RAG			8003	20 1976 INIT 1	0..... 0	
207			20	1976	I	NIT 1			
		(8003)	TRD	M	I	NIT 1		0 0	en M
208	I	NIT 1	SAG	I	NIT 2		00 M-1940 0000	20 1940 INIT 1	
209			GNN		D	EBUT			M-1940 ≠ 0 ?
210			AAG			8003	20 M-1 INIT 1	20 1939 INIT 1	Transformation
211			20	1939	I	NIT 1			de l'ordre de R.
		(8003)	TRD	M-1	I	NIT 1			
212	I	NIT 2	20	1940	I	NIT 1			Constante

= Code - 03 U V : remise à zéro de la mémoire U

						0...0T0003	U V 00	
214	T	0003	SAG	8003		0.....0		
215			E DI		D	ADSB		21 0000 DEBUT
216			21	0000	D	EBUT		

.../...

Les ordres DADSB = 1679 (n° 197) et SBFW1 = 1825 (n° 198) déjà utilisés pour l'ordre - 01 U V permettent de former dans le distributeur l'ordre 21 U DEBUT qui s'exécute alors, envoyant le contenu de 8003, c'est-à-dire zéro, dans la mémoire U.

= Code - 04 U V : entrée dans le distributeur du contenu de U et arrêt par l'ordre 01 U xxxx. Aller ensuite à V.

Cet ordre est utile pour contrôler un résultat en cours de calcul sans faire d'arrêt prédéterminé.

218	T 0004	D AD	0002	0..0T0004	U V 00		
219		E DI	R G0 W2	0...0 xx	xx U V	60 0000 W0002	
220		S BF	W 0001			60 U W0002	en W0001
221		E DI	A RDEB			01 0000 DEBUT	
222		S BF	W 0002	W 0001		01 U DEBUT	en W0002
	(W 0001)	R AG	U	W 0002	0.....0	(U)	(U)
	(W 0002)	A RT	U	D EBUT			

- L'ordre d'entrée dans le distributeur est ici un RAG au lieu de EDI. La seule raison est que cela permet d'utiliser la constante 60 0000 1992 contenue dans la mémoire RG0 W2 = 1810 (n° 167) qui est utilisée une autre fois dans la partie interprétative. Il est donc inutile de créer une constante supplémentaire.
- Cette partie de programme pourrait comporter un ordre de moins. Il suffirait pour cela de former l'ordre 01 U DEBUT avant l'ordre 60 U W0002. Le dernier ordre de substitution serait alors SBF W 0001 W 0001 ou SBF W0001 8001 c'est-à-dire l'ordre contenu en SBFW1 = 1825 (n° 198) et déjà utilisé. Une mémoire serait alors libérée. Cette amélioration sera apportée ultérieurement au C. D. P.

= Code - 05 U V : Fin du langage machine à U.

Le dernier ordre en langage machine doit avoir comme adresse instruction suivante : 1800, c'est-à-dire DEBUT. Le programme continuera alors à l'ordre en code placé en V.

Le seul ordre à former par la partie interprétative est 00 xxxx U c'est-à-dire : aller à U. En effet, le programme en langage machine commençant à U se

déroule jusqu'à ce qu'une adresse instruction suivante 1800 se présente. A ce moment, l'adresse V se trouve dans la mémoire MEMNN (placée par la partie interprétative commune à tous les ordres à 1 mémoire), et le programme continue comme souhaité.

Le seul ordre à écrire est donc :

195	T 0005	D AD	0006	8002	0...0T0005	U	V	00
	(8002)	S ØP	xxxx	U	0..... 0	00	T0005	U

Lorsqu'on est sûr de n'utiliser un programme qu'en logique extérieure, on peut exécuter des tests et des aiguillages à l'intérieur du programme en langage machine et prévoir des rentrées en codes à des adresses différentes. Par exemple, si la rentrée pour une branche doit se faire en W, il suffit que cette branche place 00 0000 W en 1884 = MEMNN puis s'adresse à 1800 = DEBUT. Mais ceci n'est pas possible en autoprogrammation, à moins de prendre de très grosses précautions.

= Code - 06 U V : Perforation simple.

Cet ordre est équivalent à l'ordre en langage machine 71 U V.

213	T 0006	E DI	P PFDEB	D DADSB	0...0 T0006	U	V	00
						71	xxxx	DEBUT

- La mémoire PFDEB = 1738 (n° 40) est aussi utilisée pour l'analyse. Elle contient 71 P 0001 DEBUT c'est-à-dire 71 1977 1800.
- En DADSB = 1679 (n° 197) la machine place U dans l'accumulateur droit en position d'adresse facteur puis fait un SBF pour former l'ordre 71 U DEBUT qu'elle exécute.

= Code - 07 U V : lecture.

Cet ordre est équivalent à l'ordre 70 U V en langage machine à condition que les cartes lues soient des cartes "lecture". En effet la partie interprétative forme l'ordre 70 U DEBUT et l'exécute. Si la carte est une carte

"lecture", elle est lue et le programme continue en DEBUT = 1800 où il commence à interpréter l'ordre V (car 00 0000 V est en MEMNN = 1884). Mais si la carte est "chargement", après la lecture le programme va à U qu'il essaye d'interpréter en langage machine. Cette propriété peut au besoin être utilisée à condition que le programme ne soit traité qu'en logique extérieure. Si ce n'est pas le cas on peut se tirer d'affaire en prenant de grosses précautions.

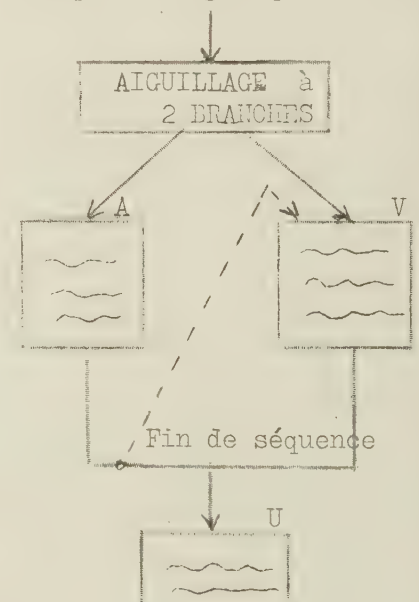
196 | T 0007 | E DI | L CDEB | D ADSB | 0...0T0007 | U V 00 | 70 0000 DEBUT |

- La mémoire LCDEB = 1726 (n° 204) contient 70 0000 1800 c'est-à-dire 70 0000 DEBUT.
- En DADSB = 1679 (n° 197), U est placé en position d'adresse facteur dans 8002; ensuite, l'ordre 70 U DEBUT est formé par un SBF puis exécuté.

= Code - 09 U V : Fin de séquence.

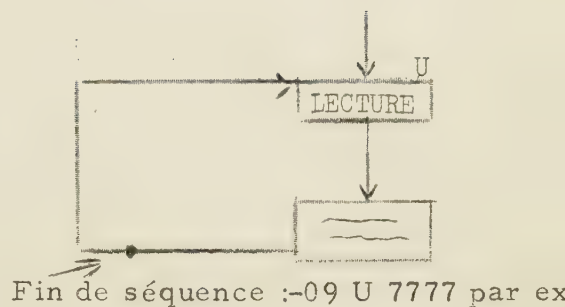
- Utilité de cet ordre : nous la montrerons sur un exemple d'organigramme.

En autoperforation, il faut parcourir successivement les 2 branches de l'aiguillage pour pouvoir perforer les programmes en langage machine correspondants. S'il n'y avait pas d'ordre de fin de séquence dans l'organigramme ci-contre, la branche A serait autoperforée mais pas la branche V. En autoperforation, il faut donc aller à V à la fin de la branche A. Par contre, en logique extérieure et en exécution de programme autoperforé, c'est à U que l'on doit aller après la branche A. L'ordre de fin de



séquence - 09 U V envoie donc à U en logique extérieure et en exécution de programme autoperforé, mais à V en autoprogrammation. Cet ordre doit en général figurer à la fin d'une des branches des tests à 2 branches et à la fin de 2 de celles des tests à 3 branches. Il doit également être utilisé lorsqu'on arrive à un endroit qui doit marquer la fin de l'autoperforation et non celle du

déroulement du programme. Par exemple :
La fin de séquence sert ici à stopper l'auto-programmation. Lorsque le programme s'exécute (en logique extérieure ou en langage machine), le programme s'arrêtera quand il n'y aura plus de cartes à lire.



D'une façon générale, la fin de séquence est utilisée lorsque les trajets suivis par la logique extérieure et l'autoperforation sont différents (sauf pour le bouclage).

- Réalisation : En logique extérieure, cet ordre est un saut inconditionnel à U.

200	T 0009	S AG	8003	0...0T0009	U V 00	
201		D AD	0006		0...0 U	
202		T RD	MEMNN	D DEBUT	0...0 U	en MEMNN = 1884

Le programme envoie U à la place de V comme adresse instruction suivante en MEMNN = 1884 puis va en DEBUT = 1800.

VII - B - 3 - Remarque sur les ordres de service :

Pour l'exécution de tous ces ordres à 1 mémoire ou ordres de service, apparaissent nettement :

- La partie commune, d'abord à tous les ordres, ensuite aux ordres de service.
Cette partie prépare tout ce qui est utile à l'ensemble des ordres de service (analyse et adresse V de l'instruction suivante) et effectue l'aiguillage à branches multiples qui différencie les ordres...
- La formation des ordres à exécuter
- L'exécution de ces ordres
- Le retour à DEBUT.

Pour les ordres à 2 mémoires, les plus fréquents et les plus complexes, apparaissent deux faits nouveaux : l'indexage et l'existence de sous-programmes.

VII - C - ORDRES A DEUX MEMOIRES.

Chaque adresse représentée par une lettre (par exemple N) sera notée indifféremment par cette lettre (N) ou par la lettre écrite autant de fois que l'adresse a de chiffres (NNN).

VII - C - 1 - Partie commune à tous ces ordres : débute en DEUME = 1709 (n°48)

a) Début : Sa principale fonction est de séparer les diverses indications contenues dans l'ordre.

48	D EUME	T RD	P 0002	0 0	PP'NNNiAAAA	PP' NNNi AAAA	en P2 (pour analyse)
49		E DI	8003			0 0	
50		S BI	P 0005			0 0 A	en P5 = 1981
51		S AD	8001		PP'NNNi0000		
52		D AG	0001	0 OP	P'NNNi00000		
53		A AG	C TEL	0...0G000P	P'NNNi0...0	00G0000	
54		T RG	W 0002			00G000P	en W2 = 1992
55		S AG	C TEL	0..... OP	P'NNNi0...0		
56		D AG	0001	0.....OPP'	NNNi 0....0		
57		S AG	8003	0 0		0OPP'	
58		T DI	P PPRI			0OPP'	en PPPRI = 1750
59		D AG	0003	0 ...ONNN	io 0		
60		T RG	M EMNN			0' 0 N	en MEMNN = 1884
61		T RD	W 0003			io 0	en W3 = 1993
62		R AD	P 0001	0..... 0	00 n 0000		
63		A AD	G J	8002	67 n+1 GL	67 0001 GL	
64	G J	67	0001	G L			
	(8002)	R AV	n+1	G L	0 0	jBBBBkCCCC	j B k C
65	G L	T DI	P 0003			j B k C	en P3 (pour analyse)
66		E DI	8003			0 0	
67		S BI	P 0008			0 00000 C	en P8 = 1984
68		S AD	8001		jBBBBk0000		

69	D AG	0001		0....0 j	BBBBk0...0	
70	S AG	8003		0..... 0		0.....0 j
71	T DI	W 0005				0.....0 j en W5 = 1995
72	D AG	0004	I INNIN	0.....0 B	k0..... 0	

- P0005 = 1981 et P0008 = 1984 servent ici de mémoires de travail.
- La mémoire CTE 1 = 1802 (n°159) contient la constante 00 0000 G 0000 c'est-à-dire 00 0000 1856. En lui ajoutant F, on forme l'ordre d'aiguillage multiple "aller à G 000 F" qui marquera la séparation des branches correspondant aux différents F. Cet ordre est mis en réserve dans la mémoire de travail W0002 = 1992 par l'ordre n°54.
- L'ordre n°60 met en place l'adresse de l'instruction suivante dans la mémoire MEMNN = 1884.
- Quand tout le parti possible a été tiré de la moitié gauche de l'instruction, c'est la partie droite qui est considérée (celle contenue en n+1). Le mot 00 0000 N a pris la place de 00 0000 n en MEMNN = 1884, mais l'adresse n se trouve encore dans la mémoire P0001 = 1977. Les ordres n°62 et 63 forment l'ordre qui va chercher la valeur absolue de la partie droite de l'instruction.
- C'est l'ordre INNIN qui enverra 00 0000 B en P0006 = 1982 utilisée comme mémoire de travail. Quant à k 0.....0, il ne sera envoyé en mémoire que si l'indexage a lieu (paquet "index" chargé).

.../...

b) Index et fin de la partie commune :

Le contenu de INNIN = 1759 (n° 73) est différent selon que le paquet supplémentaire "index" est chargé ou non. Si ce paquet n'est pas chargé, INNIN = 1759 contient : TRG P0006 NINCH = 21 1982 1803. Cet ordre envoie donc à NINCH = 1803 (n° 74) : voir organigramme. Si ce paquet est chargé, INNIN = 1759 contient : TRG P0006 INCHA = 21 1982 1785. Cet ordre envoie à INCHA = 1785 (n° 226).

= Cas où le paquet "index" n'est pas chargé :

73	I NNIN	T RG	P 0006	N INCH	0.....0 B	k0..... 0	0.....0 B	en P6 = 1982
74	N INCH	R AG	P 0008	C ØMMU	0.....0 C	0..... 0		
75	C ØMMU	D AG	0004		00 C 0000			
76		A AG	0 0ANA		00 C ANALY			
77		T RG	P 0007	W 0002			00 C ANALY	en P7 = 1983
	(W 0002)	S ØP	0000	G 000P				Aiguillage

La mémoire P0007 = 1983 est utilisée ici comme mémoire de travail.

= Cas où le paquet "index" est chargé :

(CTR_i) désigne le contenu du registre d'index n°i.

73	I NNIN	T RG	P 0006	I NCHA	0.....0 B	k0..... 0	0.....0 B	en P6 = 1982
226	I NCHA	T RD	W 0007				k0..... 0	en W7 = 1997
227		R AG	P PPRI		0....OPP'	0 0		
228		S AG	0 080		0..OPP+80			
229		A NG	I NF	W 0002				P ≠ 8 et 9 ?
230	I NF	R AD	W 0003		0..... 0	i0 0		Cas où P < 8
231		D AD	0005			00000i0000		
232		A AD	R DCKA	8002		65CTR _i KA		
	(8002)	R AD	C TR _i	K A		0...0(CTR _i)		

233	K A	A AG	8001	0...0 (CTR _i)	0....0 (CTR _i)	
234		A AG	P 0005	0..... 0A'		0.....0A
235		T RG	P 0005			0.....0 A' en P5 = 1981
236		D AG	0003		0...0(CTR _i)000	
237		T RD	P 0004			0...0(CTR _i) 000 en P4 = 1980
238		R AD	W 0005	0 0	0 0 j	
239		D AG	0004		0....0 j 0000	
240		A AD	R DCKB 8002		65 CTR _j KB	
	(8002)	R AD	C TR _j K B	0..... 0	0....0 (CTR _j)	
241	K B	A AG	8001	0...0 (CTR _j)		
242		A AG	P 0006	0..... 0 B'		0..... 0 B
243		T RG	P 0006			0.... 0 B' en P6 = 1982
244		A AD	P 0004		0000(CTR _i)(CTR _j)	
245		D AG	0003		0(CTR _i)(CTR _j)000	
246		T RD	P 0004			0(CTR _i)(CTR _j) 000 en P4 = 1980
247		R AD	W 0007	0 0	k0 0	
248		D AD	0005		0....0k0000	
249		A AD	R DCKC 8002		65 CTR _k KC	
	(8002)	R AD	C TR _k K C	0 0	0....0(CTR _k)	
250	K C	A AG	8001	0 ...0(CTR _k)		
251		A AG	P 0008	0 0 C'		0 0 C
252		A AD	P 0004		0(CTR _i)(CTR _j) (CTR _k)	
253		S AD	8002		0 0	0(CTR _i)(CTR _j) (CTR _k)
254		T DI	P 0004			0(CTR _i)(CTR _j) (CTR _k) en P4 = 1980
255		T RG	P 0008 C ØMMU	0 0 C'	0 0	0 0 C' en P8 = 1984
256	R DCKA	R AD	C 0000 K A			
257	R DCKB	R AD	C 0000 K B			
258	R DCKC	R AD	C 0000 K C			

- Les instructions n° 227, 228, 229 permettent d'effectuer le test :
"P $\neq$ 8 et 9 ?". En fait, c'est le test "P $<$ 8 ?" qui est exécuté, ce qui revient au même. Si P est égal à 8 ou 9, le programme va directement à W 0002 = 1992 où se trouve l'ordre d'aiguillage 00 0000 G 000P = 00 0000 G 0008 ou 00 0000 G 0009 (voir organigramme). Si P est différent de 8 et 9, le programme va en INF = 1642 (n° 230) pour exécuter l'indexage.
- La mémoire "0080" = 1822 (n° 161) contient la constante 00 0000 0080.
- Les ordres n° 230, 231, 232 servent à former l'ordre 65 CTRi KA = 65 CTRi 1651 qui fait entrer dans l'accumulateur droit le contenu du registre d'index n° i. Les ordres n° 238, 239, 240 jouent le même rôle pour l'index j, et les ordres n° 247, 248, 249 pour l'index k.
- L'ordre n° 235 envoie $A' = A + (CTRi)$ à la place de A, c'est-à-dire en P 0005 = 1981. De même les ordres n° 243 et n° 255 envoient respectivement B' et C' dans les mémoires P 0006 = 1982 et P 0008 = 1984.
- La partie de programme "indexage" sert également à former le mot 4 de l'analyse, c'est-à-dire
$$0 \text{ xxx xxx xxx}$$
$$(CTRi)(CTRj)(CTRk)$$

Ce mot est formé petit à petit, au fur et à mesure qu'interviennent les 3 registres d'index. L'ordre n° 254 envoie ce mot en P 0004 = 1986 (4ème mot de la zone de perforation).

- En CØMMU = 1789 (n° 75), le programme correspondant au cas où le paquet "index" est chargé rejoint le programme sans paquet "index".

Remarque : l'assemblage de cette partie de programme a été fait de telle façon qu'elle remplisse une séquence compacte (1639 à 1670). Cet assemblage a donc été fait à la suite de l'assemblage du reste du programme, séparé de celui-ci par le passage d'une carte PBD 1639 1670 et d'une carte PRA 1671 1999 (de sécurité).

= En CØMMU = 1789 (n° 75), la machine forme 00 C ANALY ou 00 C' ANALY (ANALY = 1700) et l'envoie en P 0007 = 1983 utilisé comme mémoire de travail à ce moment là. Ensuite, le programme arrive en W 0002 = 1992 où se trouve l'ordre d'aiguillage 00 0000 G 000P = 00 0000 1856 + P, placé par l'ordre n° 54. Dans la suite du texte, nous désignerons par A l'adresse A indexée ou non (suivant que le paquet "index" est ou non chargé) c'est-à-dire A ou A'. Cela ne risque pas de prêter à confusion car, lorsque A' est créé, A n'a plus aucune importance. La même remarque est applicable à B et C.

c) Analyse

La partie de programme relative à l'analyse de la plupart des ordres à 2 mémoires commence par un test sur le signe du pupitre. Si ce signe est -, l'analyse doit être faite. Les mémoires P 0001, P 0002, P 0003, P 0004 sont déjà pourvues quand on arrive en ANALY = 1700 (n° 23).

.../...

Il reste donc à envoyer (A) en P 0005 = 1981, (B) en P 0006 = 1982, (C) en P 0007 = 1983 et (C + 1) en P 0008 = 1984.

23	A	NALY	R	AG	8000						
24			A	NG	A	NA	D	EBUT			Analyse ?
25	A	NA	R	AG	P	0005		0....0	A	0.....0	
26			A	AD	P	0006				0..0	B
27			D	AG		0004		00	A	0000	00 B 0000
28			A	AG	E	DOKM	8003	69	A	KM	69 0000 KM
		(8003)	E	DI		A				K M	(A)
29	K	M	T	DI	P	0005					(A)
											en P5 = 1981
30			A	AD	E	DOKN	8002		69	B	KN
		(8002)	E	DI		B				K N	(B)
31	K	N	T	DI	P	0006					(B)
											en P6 = 1982
32			R	AD	P	0007	S	UANA	0		0
33	S	UANA	E	DI	E	DOKP					00 C ANALY
											69 xxxx KP
34			S	BF	E	DOKP	8001				69 C KP
		(8001)	E	DI		C				K P	(C)
35	K	P	T	DI	P	0007					(C)
											en P7 = 1983
36			A	AD	O	10					OOC+1ANALY
37			E	DI	E	DOKR					69 xxxx KR
38			S	BF	E	DOKR	8001				69 C +1 KR
		(8001)	E	DI		C+1				K R	(C+1)
39	K	R	T	DI	P	0008	P	FDEB			(C+1)
											en P8 = 1984
40	P	FDEB	P	F	P	0001	D	EBUT			

- La mémoire EDOKM = 1701 (n° 180) contient 69 0000 KM = 69 0000 1751

" EDOKN = 1737 (n° 181) contient 69 0000 KN = 69 0000 1801

" EDOKP = 1690 (n° 182) contient 69 xxxx KP = 69 xxxx 1702

" EDOKR = 1696 (n° 185) contient 69 xxxx KR = 69 xxxx 1752

- La mémoire O10 = 1689 (n° 160) contient 00 0001 0000

.../...

VII - C - 2 - P = 0 : sous-programmes à 2 adresses - Opérations virgule fixe -

Tests -

Dès le début de cette partie, les programmes relatifs aux différentes valeurs de P' se séparent. Pour cela, la machine forme l'ordre d'aiguillage $00\ 0000\ F\ 00PP' = 00\ 0000\ F\ 000P' = 00\ 0000\ 1846 + P'$. Ce test est aussitôt exécuté.

149	G 0000	R AG	P PPRI		0..... 0 PP'	0 0	
150		A AG	C TE2	18003	0.... 0 F 00PP'		0....0 F 0000
	(8003)	S ØP	0000	F 000P'			

- La mémoire CTE 2 = 1771 (n° 177) contient $00\ 0000\ F\ 0000 = 00\ 0000\ 1846$
- Dans la mémoire PPPRI = 1750, l'ordre n° 58 a envoyé $00\ 0000\ 00PP'$
- Remarque : Les mémoires F 0000, F 0001, ..., F 0008 contiennent les premiers ordres des branches relatives respectivement à $PP' = C1, C2, \dots, C8$. La mémoire F 0009 est libre, ce qui permettra de créer sans difficulté un ordre $PP' = C9$.

a) PP' = 00 : sous-programmes à 2 adresses :

La partie de programme commençant à F 0000 = 1846 forme les ordres d'entrée dans les sous-programmes à 2 adresses. Ceux-ci servent en général à calculer des fonctions. Leur standardisation est assez délicate. Certains des sous-programmes existant demandent que l'argument soit en 8002, d'autres le veulent en 8003, d'autres enfin dans les deux accumulateurs.

.../...

Cela ne permet pas de faire une entrée standard. Pour standardiser l'entrée, que l'argument soit en simple ou double précision, il faut obligatoirement faire figurer dans l'accumulateur non pas l'argument lui-même mais son adresse. Cette adresse servira à former le ou les ordres destinés à entrer l'argument dans l'accumulateur. Cette entrée sera le plus souvent faite par un ordre de RAD. C'est donc sous la forme 65 B xxxx que le C.D.P. envoie l'adresse B dans l'accumulateur. Il met de plus en position "adresse instruction suivante" de cet ordre l'adresse A + 1 qui est une adresse du sous-programme. L'ordre 65 B A + 1 est donc directement utilisable et exécutable. Le C.D.P. le place en 8003. L'ordre de sortie est placé par le C.D.P. en 8002 et non en 8001 pour pouvoir, si besoin est, le transformer plus facilement.

En résumé, l'entrée pratiquée par le code est donc de la forme :

Adresse	C Ø	F	I S	(8003)	(8002)
	RAG	X	Ø	65 B A + 1	0 0
X	AAD	(	A		20 C N'
α	65	B	A + 1		
β	20	C	N'		

Les sous-programmes employés doivent utiliser cette disposition.

Remarque 1 : 20 C N' est l'ordre de sortie ou tout au moins servira à former celui-ci. N' est l'adresse de la 1ère instruction en langage machine à effectuer après le sous-programme. Si l'on est en logique extérieure, la 1ère instruction est l'ordre ANALY = 1700 (n° 23) (voir l'organigramme). On a donc N' = 1700 = ANALY.

.../...

Si l'on a par contre autoperforé le programme et que on l'exécute en langage machine, la 1ère instruction après le sous-programme est l'instruction N. On a donc $N' = N$.

Remarque 2 : La forme de l'entrée pratiquée par le code qui est indiquée ci-dessus est schématisée. En fait, les mots α et β sont directement formés dans les accumulateurs, du moins en logique extérieure. Dans ce cas, $20 C N'$ n'apparaît pas dans le distributeur.

				0..0xxxx	0 0		
151	F 0000	A AD	P 0006		0 0B		
152		D AG	0004		00 B 0000		
153		A AD	P 0005		00 B A		
154		E DI	K E			15 P7 xxxx	
155		S BI	K E			15 P7 A	en KE
156		R AG	8002	00 B A	0 0		
157		A AG	R D01	<u>65 B A+1</u>		6500000001	
158		A AD	T D00	K E	20 0	20 0	
	(K E)	A AD	P 0007	<u>A</u>	<u>20 C ANALY</u>	<u>00 C ANALY</u>	

- L'adresse A en position d'adresse instruction dans l'accumulateur sert d'une part à former l'ordre 65 B A+1 et d'autre part à former l'ordre d'entrée dans le sous-programme : AAD P0007 A.

- La mémoire KE = 1821 (n° 170) contient la constante 15 P0007 xxxx = 15 1983 xxxx.

- La mémoire RD01 = 1843 (n° 179) contient 65 0000 0001.

- La mémoire TD00 = 1868 (n° 171) contient 20 0000 0000.

- La mémoire P0007 = 1983 contient 00 C ANALY = 00 C 1700 envoyé par l'ordre n° 77.

.../...

b) PP' = 01 : addition en virgule fixe .

- Les parties de programme relatives aux opérations en virgule fixe ont été construites de façon à avoir entre elles le plus possible de parties communes.

- Les ordres que construit la partie relative à l'addition sont :

	RAG	A	W0002
W0002	AAG	B	W0003
W0003	TRG	C	ANALY

- La formation des ordres RAG et TRG, pouvant avoir une partie commune avec d'autres opérations, a été placée après la formation de l'ordre AAG.

79	F 0001	R AD	O DIXO	A DSØ1	0 0	00 0010 0000	
81	A DSØ1	A AD	P 0006			00 0010 B	0 0 B
82		D AG	0004			10 B 0000	
83		A AD	O OW3			10 B W0003	0...0 W0003
84		T RD	W 0002			10 B W 0003	en W2
85		R AG	P 0005		0 0 A	0 0	
86		D AG	0004		00 A 0000		
87		A AD	T GOO	K G		210 0	
88	K G	A AG	R GOW2		60 A W0002		60 0000 W0002
89		A AD	P 0007			21 C ANALY	00 C ANALY
90		T RD	W 0003	8003		21 C ANALY	en W3
	(8003)	R AG	A	W 0002	(A)	0 0	
	(W 0002)	A AG	B	W 0003	(A) + (B)		
	(W 0003)	T RG	C	A NALY		(A) + (B)	en C

.../...

- La mémoire ODIKO = 1754 (n°164) contient la constante 00 0010 0000
- Pour former 10 B W0003 on aurait pu amener d'abord 0.....0 B dans l'accumulateur droit, faire un décalage de 4 positions à gauche et ajouter 10 0000 W0003. Il y aurait eu 3 ordres au lieu de 4. Mais la partie relative à la soustraction n'aurait pu rejoindre celle de l'addition qu'après ces 3 ordres, et en en comportant aussi 3, ce qui fait 6 ordres en tout. Or, dans le cas présent (voir § c) la partie "soustraction" rejoint la partie "addition" en ADSØ1 = 1760 (n°81) après un seul ordre. L'ensemble a donc 4 + 1 = 5 ordres. Les 2 parties étant séparées au début, on a gagné de la place en écrivant en premier les ordres non communs.
- La mémoire 00 W3 = 1753 (n°165) contient 00 0000 W0003 = 00 0000 1993
- La mémoire T G 00 = 1748 (n°166) contient 21 0..... 0
- La mémoire RGOW2 = 1810 (n°167) contient 60 0000 W0002 = 60 0000 1992

c) PF' = 02 : soustraction en virgule fixe .

- Les ordres que construit la partie relative à la soustraction sont :

	RAG	A	W0002
W0002	SAG	B	W0003
W0003	TRG	C	ANALY

La seule différence avec les ordres d'addition est le SAG.

La seule partie du programme particulière à la soustraction est :

80 | F|0002 | R|AD | 0|ØNZO | A|DSØ1 | 0 0 | 00 0011 0000 | 00 0011 0000 |

La mémoire ØØNZO = 1804 (n°184) contient 00 0011 0000.

d) IPF' = 03 et IPF' = 05 : multiplication en virgule fixe sans ou avec décalage.

- Les ordres que construit la partie relative à la multiplication sans décalage pourraient être :

	RAG	A	W0002
W0002	MUL	B	W0003
W0003	TRD	C	ANALY

Mais le produit (A) x (B) risque d'avoir 20 chiffres, c'est-à-dire que le résultat peut dépasser la capacité de l'accumulateur droit. Dans ce cas, la simple précision n'est plus respectée et il faut que la machine le signale pour que l'on ne risque pas d'envoyer dans C une valeur qui ne serait que la partie droite du résultat. On fait donc pour cela un test sur la nullité de l'accumulateur gauche après la multiplication. Les ordres à former sont donc de la forme :

	RAG	A	W0002	
W0002	MUL	B	KF	
KF	GNN	ARETH	W0003	= 44 1815 1993
W0003	TRD	C	ANALY	
ARETH	ART	1333	W0003	= 01 1333 1993

Ce n'est pas encore tout à fait ces ordres là que l'on forme.

En effet, on a ajouté un ordre pour que ce programme se rapproche plus de celui relatif à la multiplication avec décalage. Pour celle-ci, les ordres que forme le C.D.F. sont : (remarquons que le décalage a lieu après la multiplication et avant le test)

.../...

	RAG	A	W0002	
W0002	MUL	B	W0001	
W0001	3 x	000m	KF	Ordre de décalage
KF	GNN	ARETM	W0003	Ordre constant
W0003	TRD	C	ANALY	
ARETM	ART	1333	W0003	Ordre constant

On a donc adopté comme ordres à former pour la multiplication sans décalage :

	RAG	A	W0002	
W0002	MUL	B	W0001	
W0001	00	0000	KF	
KF	GNN	ARETM	W0003	Ordre constant
W0003	TRD	C	ANALY	
ARETM	ART	1333	W0003	Ordre constant

Programme relatif à la multiplication sans décalage :

92	W0003	TRD	W0001	0....0 XXXX	0 0	0 0	en W1 = 1991
93		RAG	P0006	0 0 B	0 0		
94		DAG	0004	00 B 0000			
95		AAG	MLOW1	19 B W1			
96		TIG	W0002		19 B W1		en W2 = 1992
97		RAD	O2KF	KI	0 C	00 0002 KF	
98	KI	EDI	W0001			0 0	
99		SBI	W0001			0 0 KF	en W1 = 1991
100		RAG	P0005	0 0 A	0 0		
101		DAG	0004	00 A 0000			
102		AAD	TDOO	KIG		20 0	
88	(K,G)	AAG	R'GOW2		60 A W0002		
89		AAD	P'0007			20 C ANALY	
90		TED	W'0003	8003		20 C ANALY	en W3 = 1993

.../...

- La mémoire KF = 1762 (n°102) contient le test GNN ARETM W003 = 01 1015 1993.
- La mémoire ARETM = 1815 (n°172) contient l'ordre d'arrêt que provoque l'apparition d'un produit de plus de 10 chiffres. Cet ordre est : 01 1333 W0003 = 01 1333 1993. Il est caractéristique de l'apparition d'un produit de plus de 10 chiffres.
- L'ordre F0003 = 1849 (n°91) prépare l'ordre KI = 1712 (n°97) en envoyant 0.....0 en W0001 = 1991. Nous verrons plus loin que la partie de programme commençant en F0005 = 1851 (n°103) prépare aussi l'ordre KI en envoyant l'ordre de décalage en W0001.
- La mémoire ML0W1 = 1811 (n°168) contient 19 0000 W0001 = 19 0000 1991.
- La mémoire 02.KF = 1806 (n°169) contient 00 0002 KF = 00 0002 176
Le 2 en position facteur ne sert à rien. Le programme sous sa forme initiale utilisait cette constante ; comme le 2 n'est pas nuisible, il a été conservé.
- La mémoire TD00 = 1808 (n°171) contient 20.....0.
- La partie commençant en KG = 1706 (n°88) fait aussi partie du programme d'addition, au cours duquel elle a été écrite.
- Après l'ordre n°90 s'exécutent l'ordre 60 A W0002 et la suite des ordres de la multiplication (2ème programme encadré).

Programme relatif à la multiplication avec décalage :

La seule partie particulière se compose des 3 ordres suivants :

103	F 0005	R AD	P 0001	0	0	00 n 0000
104		A AD		8002		65 n+2 F0003 65 0002 F0003
105		65	0002	F 0003		
	(8002)	R AD	n +2	F 0003	0	0 3x 000m 1939

Ces ordres servent à entrer dans l'accumulateur droit l'ordre de décalage placé en $n + 2$. L'ordre F0003 = 1049 (n°91) envoie ensuite cet ordre en W0001 = 1991. La seule différence, dans la suite, avec la multiplication sans décalage est que c'est l'ordre 3x 000m KF qui est formé en W0001 = 1991, et non 00 0000 KF. C'est donc le 1er programme encadré qui sera exécuté après l'ordre n° 90.

c) FF' = 04 et 06 : division en virgule fixe sans ou avec décalage :

Les ordres que construit la partie relative à la division sans décalage pourraient être :

	RAD	A	W0002
W0002	DRR	B	W0003
W0003	TRD	C	ANALY

Pour permettre au programme relatif à la division de rejoindre plus tôt celui de la multiplication, et en particulier avant l'ordre KG, ce n'est pas RAD A W0002 mais RAG A W0002 qui est formé comme 1er ordre. Cela oblige à ajouter un ordre qui paraît artificiel mais permet d'économiser de la place.

.../...

	RAG	A	W0002
W0002	RAD	8003	W0001
W0001	DRR	B	W0003
W0003	TRD	C	ANALY

Mais là n'est pas encore ce qui a été adopté. En effet, pour permettre une plus grande similitude entre les divisions avec et sans décalage, on a encore ajouté un ordre.

Pour la division avec décalage, les ordres formés sont :

	RAG	A	W0002
W0002	RAD	8003	W0001
W0001	3x	00Cm	KH
KH	DRR	B	W0003
W0003	TRD	C	ANALY

Décalage (placé avant la division)

Par analogie, les ordres formés pour la division sans décalage sont :

	RAG	A	W0002
W0002	RAD	8003	W0001
W0001	00	0000	KH
KH	DRR	B	W0003
W0003	TRD	C	ANALY

.../...

Programme relatif la division sans décalage :

106	F 0004	T RD	W 0001	0 ... 0 xxxx	0 0	0 0	en W1 = 1991
107		R AD	P 0006	0 0	0 0 B		
108		D AG	0004		00 B 0000		
109		E DI	K H			64 xxxxW0003	
110		S BF	K H			64 B W0003	en KH = 1716
111		E DI	R DAW1			65 8003W0001	
112		T DI	W 0002			65 8003W0001	en W2 = 1992
113		R AD	O 2KH	K I	0 0	00 0002 KH	
97	(K I)	E DI	W 0001			0 0	
98	()	S BI	W 0001			0 0 K H	en W1 = 1991
99	()	R AG	P 0005	0 0 A	0 0		
100	()	D AG	0004	00 A 0000			
101	()	A AD	T D00	K G	20 0		
88	(K G)	A AG	R GOW2	60 A W0002			
89	()	A AD	P 0007		20 C ANALY		
90	()	T RD	W 0003	8003		20 C ANALY	en W3 = 1993

- L'ordre F0004 = 1853 (n°106) prépare l'ordre KI = 1712 (n°97) en envoyant 0...0 en W0001 = 1991. Nous verrons plus loin que la partie de programme commençant en F0006 envoie, elle, l'ordre de décalage en W0001.

- La mémoire KH = 1716 (n°173) contient 64 xxxx W0003 = 64 xxxx 1993.

- La mémoire RDAW1 = 1672 (n°174) contient 65 8003 W0001 = 65 8003 1991.

- En KI = 1712 (n°97) le programme relatif à la division rejoint celui relatif à la multiplication. Le nombre d'ordres particuliers

.../...

La division est donc assez restreint; ceci est dû surtout au fait que l'on ait pu se ramener à avoir comme 1er ordre : 60 A W0001.

- Après l'ordre n°90 s'exécutent les ordres du 2ème programme encadré.

Programme relatif à la division avec décalage :

La seule partie particulière à la division avec décalage se compose de :

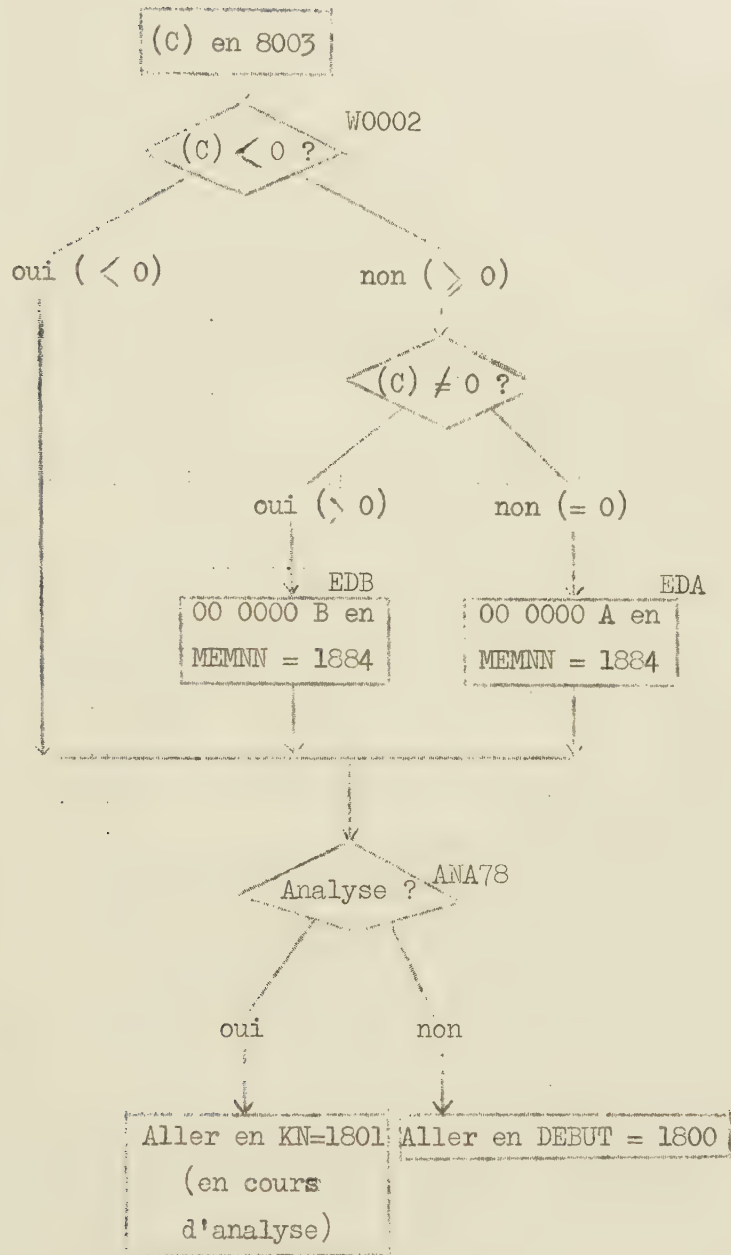
114	F' 0006	R AD	P 0001	0	0	00 n 0000	
115		A AD		8002		65 n+2 F0004	65 0002 F0004
116		65	0002	F 0004			
	(8002)	R AD	n+2	F 0004	0	0	3x 000m 1991

Ces ordres servent à entrer dans l'accumulateur droit l'ordre de décalage placé en $n + 2$. L'ordre F0004 = 1050 (n°106) l'envoie ensuite dans la mémoire W0001 = 1991 pour préparer l'ordre KI = 1712 (n°97). La seule différence avec la division sans décalage est que c'est l'ordre 3x 000m KH et non 00 0000 KH qui est formé en W0001. C'est donc les ordres du 1er programme encadré qui sont exécutés après l'ordre n°90.

f) PF' = 07 : Test de signe à 3 branches.

Le programme que forme la partie interprétative pour exécuter le test SI peut être représenté par le schéma suivant :

.../...



Nous verrons au paragraphe X - E - 2 - d - α qu'à cause de l'exécution en langage machine, seule l'adresse C est indexable.

.../...

La branche correspondant à $(C) < 0$ doit envoyer à l'instruction d'adresse N en code . Pour cela, il faut que 00 0000 N soit placé en MEMNN = 1884, ce qui est déjà réalisé par le début de la partie interprétative. Les branches correspondant à (C) nul et à (C) positif doivent envoyer respectivement aux instructions en code contenues en A et B. Elles placent donc respectivement 00 0000 A et 00 0000 B en MEMNN = 1884.

Les ordres que forme la partie interprétative sont les suivants :

	R AG	C	W 0002	(C)	C	0	
W 0002	A NG	ANA78					$(C) < 0 ?$
	G NN	EDB	EDA				$(C) \neq 0 ?$
EDB	E DI	P0006	EDBA			0 0B	Cas où $(C) > 0$
EDA	E DI	P0005	EDBA			0 0A	Cas où $(C) = 0$
EDBA	T DI	MEMNN	ANA78			0 0B 0 0A	} en MEMNN = 1884
ANA78	R AG	8000					
	A NG	KN	DEBUT				

Pourquoi envoyer en ANA 78 et non pas en ANALY ? Nous verrons que le programme relatif au code PP' = 00 (test suivant la position B de la mémoire C) a la même fin que le programme relatif à PP' = 07. Or, pour PP' = 03 il est dangereux d'amener dans le distributeur le contenu de la mémoire B. B est en effet compris entre 0 et 9; il se peut que, par suite de la lecture d'une carte incomplète, certaines des mémoires d'adresse comprise entre 0 et 9

.../...

soient vides, ce qui provoquerait une "erreur distributeur". L'analyse des ordres 07 et 08 ne comporte donc pas l'envoi du contenu de la mémoire B dans la mémoire de perforation P 0006. Elle ne commence donc qu'à l'envoi de (C) en P 0007 = 1983. En fait, c'est à l'ordre précédent qu'elle commence, c'est à dire à l'ordre KN = 1801 (n° 31). Il ne faut par conséquent pas s'adresser à ANALY = 1700 (n° 23) qui envoie à ANA, mais à ANA 78 = 1717 (n° 127) qui envoie à KN.

Programme interprétatif relatif au test SI

117	F 0007	R AD	F 7F8	0	0	46 ANA78 xxxx	
118		46 A NA78					
119		G NN E DB	E DA				
120	E DB	E DI P 0006	E DBA			0	OB
121	E DA	E DI P 0005	E DBA			0	OA
122	E DBA	T DK M EMNN				0	A
						0	B
127	A NA78	R AG 8000					
128		A NG K N	D EBUT				
123	F 7F8	T RD W 0002				46 ANA78 xxxx	46 ANA78 xxxx en W0002 = 1992
124		R AD P 0008		0	0	0	OC
125		D AG 0004				00 C 0000	
126		A AD R GOW2	8002			60 C W0002	
	(8002)	R AG C	W 0002	(C)	0	0	
	(W 0002)	A NG A NA78					
	()	G NN E DB	E DA				

.....

.../...

.....

La succession de ces ordres, due à l'écriture en PASØ, n'est pas celle de l'exécution. En effet, après l'ordre F 0007, c'est l'ordre F 7 F 8 qui s'exécute. L'exécution effective de l'ordre SI, indiquée précédemment, débute après l'ordre n° 126.

g) PP' = 08 Test sur l'existence d'un 8 ou d'un 9 en position n° B de la mémoire C

Comme pour le test SI, seule l'adresse C est indexable à cause de l'exécution en langage machine. Les ordres que forme la partie interprétative pour l'exécution du test SD sont :

RAG	C	W 0002
W0002	SDB	ANA78 EDA

Le test 9 B s'effectue sur le distributeur. Le 1er ordre pourrait donc être EDI C W0002. Mais, pour permettre une plus grande partie commune avec l'ordre SI, c'est l'ordre RAG C W 0002 qui a été formé.

Si la réponse au test OUI, c'est-à-dire s'il y a un 8, le programme va en ANA78 = 1717 (n° 127). Le mot 00 0000 N se trouve en effet en MEMNN = 1884, c'est-à-dire que l'instruction suivante en code est bien N. Si la réponse est NON, c'est-à-dire s'il y a un 9, le programme va en EDA et 00 0000 A est envoyé en MEMNN = 1884. S'il n'y a ni 8 ni 9 en position B, aucune réponse au test n'est donnée et la machine s'arrête avec au registre programme :

9 B ANA78 EDA = 9 B 1717 1722.

Remarque : B = 0 désigne la position n° 10.

.../...

Programme interprétatif relatif au test SD :

129	F 0008	R AG	P 0006	0 0B	0 0	
130		D AD	0002	0 0	0B0 0	
131		A AD		F 7F8	9B ANA78 EDA	90 ANA78 EDA
132		90	A NA78	E DA		
123	(F 7F8)	T RD	W 0002			9B ANA78 EDA en W0002 = 1992
124	()	R AD	P 0008	0 0	0 0C	
125	()	D AG	0004		00 C 0000	
126	()	A AD	R GOW2	8002	60 C W0002	
	(8002)	R AG	C	W 0002	(C) 0 0	(C)
	(W 0002)	S DB	A NA78	E DA		

Seuls les 4 premiers mots (n° 129 à 132) sont particuliers au test SD.

Les suivants sont communs aux tests SD et SI.

VII - C - 3 - F = 1, 2, 3, 4, 5 : sous-programmes à 3 adresses.

Standardisation : Comme pour la plupart des sous-programmes à 3 adresses déjà existants (par exemple les opérations en virgule flottante), l'entrée dans le sous-programme se fait dans les conditions suivantes :

(8003)	(8002)	(8001)
0 0	00 A B	Ordre de sortie

L'ordre de sortie que forme le CDP est 20 C N'. En logique extérieure, N' = ANALY = 1700. En autoprogrammation, N' = N, adresse de l'instruction suivante en code.

.../...

Comme il a été dit au paragraphe VI - C - 2 - a - 5, les branches relatives à $P = 1, 2, 3, 4, 5, 6, 7$ se rejoignent tout de suite. En fait, nous ne nous occuperons que des P inférieurs à 6 car $P = 6$ et $P = 7$ ont été supprimés. Les mémoires $G 0001 = 1857$, $G 0002 = 1858, \dots, G 0005 = 1861$ contiennent le même mot; c'est le 1er ordre de la partie interprétative particulière aux sous-programmes à 3 adresses. Les mémoires $G 0006 = 1862$ et $G 0007 = 1863$ sont libres.

Programme interprétatif relatif aux sous-programmes à 3 adresses :

1	G 0001	P AG	P PPRI	G C/MU					
2	G 0002	P AG	P PPRI	G C/MU					
3									
37	G 0005	P AG	P PPRI	G C/MU	0 0	0 0			
4	G 0004	P AG	P 0004		0 ... 0 PP' 0000				
13		P AG	P DEL	8003	15 DEL + PP' KJ		15 DEL KJ		
	(8003)	P AD	P DEL + PP'			0 ... 0 $\Delta_{PP'}$			
42	K J	P DI	P EDP7				11 8003 xxxx		
43		P BI	P EDP7				11 8003 $\Delta_{PP'}$	en EDP7 = 1770	
44		P AD	P 0005		0 0	0 0A			
45		P AG	P 0004			00 A 0000			
46		P AD	P 0006			00 A B			
47		P AG	P 0007		00 C ANALY				
48		P AG	P 8003	P EDP7	20 C ANALY				
	(E DP7)	P AG	P 8003	P PPRI	0 0	00 A B	20 C ANALY		

Les ordres n° 133 à 143 servent à former le dernier ordre, EDP7, de ce programme interprétatif. EDP7 doit effectuer l'entrée dans le sous-programme, c'est-à-dire envoyer en $\Delta_{PP'}$. La valeur de $\Delta_{PP'}$ est indiquée dans la mémoire $1875 + PP'$ que nous appellerons aussi DEL + PP'. Les limites de cette table sont : DEL + 10 et DEL + 59 (en fait la table a été prolongée jusqu'à DEL + 61). Cette région sera

.../...

appelée en PASØ : D 0000 à D 0051. Cette dénomination est donnée pour que l'on puisse faire une réservation régionale :

PRG	D 1886	D 1939	(région D un peu prolongée)
PEN	D 0000	1885	

- Les ordres n° 140 et 141 forment l'ordre $\Delta AD \text{ DEL} + PP' K J$ qui amène Δ_{pp} , dans l'accumulateur droit. L'ordre n° 143 envoie l'ordre SAG 8003 Δ_{pp} en EDP7 = 1770.
- La mémoire ADDEL = 1730 (n° 175) contient la constante 15 DEL KJ = 15 1875 1817.
- La mémoire EDP7 = 1770 (n° 176) contient la constante 11 8003 xxxx.

VII - C - 4 - P = 9 : bouclage.

- Le programme de bouclage a été écrit sous forme d'un sous-programme. Ce sous-programme n'est pas standard et est d'ailleurs différent en logique extérieure et en exécution en langage machine. Il a été assemblé dans une séquence de mémoires commençant à 0000, puis mis sous forme translatable. En logique extérieure, son facteur de translation, Δ_{bou} , a été fixé à 1596 et placé sous la forme 00 0000 Δ_{bou} dans la mémoire DEL + 61 = D 0051 où se trouve donc l'ordre "aller à Δ_{bou} ". Le sous-programme de bouclage va chercher les différentes indications dont il a besoin dans les mémoires où les a mises la partie interprétative.

.../...

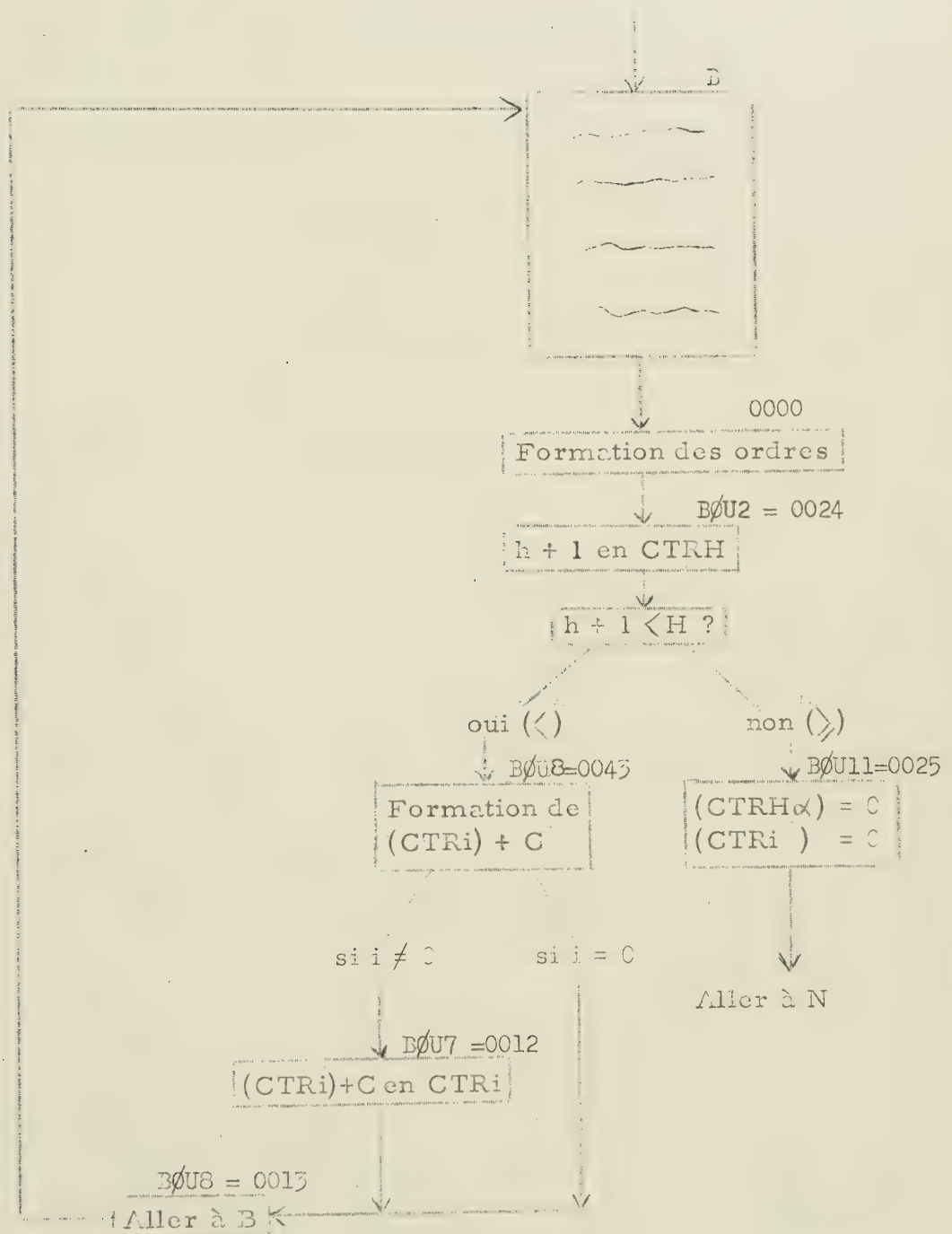
- La seule instruction particulière au bouclage avant l'entrée dans le sous-programme est :

8 | G:0009 | S:ØP | : 0000 | D:0051 | = 00 0000 1936
 | (D:0051) | S:ØP | : 0000 | Δ_{bou} | = 00 0000 1596

Pour que la 1ère instruction du sous-programme de bouclage soit bien en $\Delta_{bou} + 0$, son adresse a été fixée à 0000 dans le programme en PASØ.

- Voici le schéma suivant lequel s'exécute le bouclage 9 i N H O B O C.

Nous appellerons CTRH_i le compteur de boucles n° i, et h son contenu. L'adresse CTRi désignera le registre d'index n° i et (CTRi) sont contenu. Le sous-programme de bouclage débute par la formation d'un certain nombre d'ordres destinés à l'exécution proprement dite du bouclage.



..../...

Il peut arriver que, par suite d'une erreur, le compteur de boucles n° α ne soit pas à zéro au 1er passage par l'ordre de bouclage, et même contienne un h supérieur ou égal à H . Dans ce cas, comme on peut le voir sur l'organigramme, la boucle n'est pas répétée. Le compteur de boucles n° α et le registre d'index n° i sont remis à zéro et le programme continue en N .

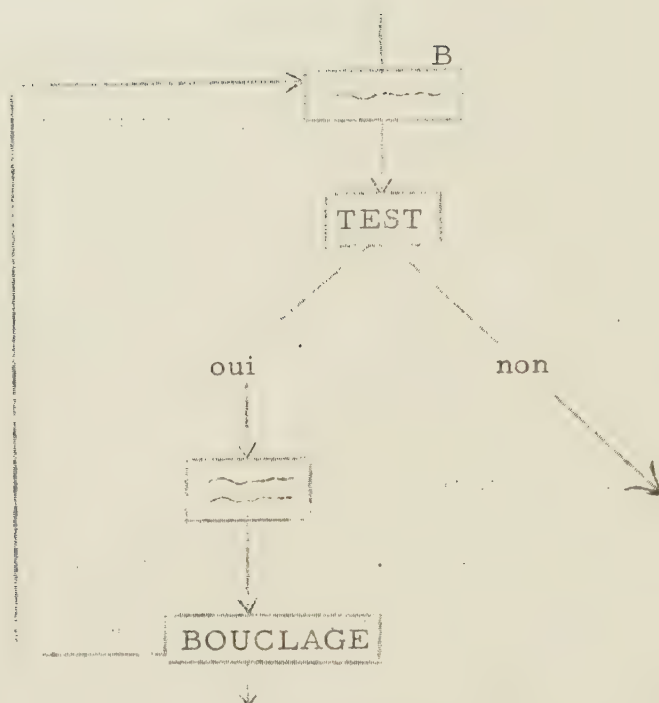
- Sauf dans des cas exceptionnels, le registre d'index n° i et le compteur de boucles n° α sont à zéro au moment où l'on passe pour la 1ère fois par l'ordre de bouclage. Chaque fois que l'on passe par cet ordre, la machine ajoute 1 au compteur de boucle et C au registre d'index. Notons que si $i = 0$, aucun registre d'index n'est modifié, quel que soit C . En effet, il serait trop dangereux de risquer de modifier le registre d'index n° 0 qui doit obligatoirement contenir zéro. Lorsque la boucle a été décrite le nombre voulu de fois, le registre d'index n° i et le compteur de boucles n° α sont remis à zéro.

- L'existence de plusieurs compteurs de boucles permet d'effectuer plusieurs boucles intérieures les unes aux autres (On ne peut pas décrire des boucles intérieures les unes aux autres en utilisant le même compteur). Lorsque le programmeur écrit des boucles intérieures les unes aux autres, il ne doit absolument pas leur attribuer le même α .

- Remarque 1 : quand une boucle ne possède pas de boucle intérieure, les ordres formés au cours du 1er passage par l'ordre de bouclage restent valables pour les autres passages et pourraient être conservés; cela permettrait d'éviter pour les autres passages la phase de formation d'ordres et ainsi de gagner du temps. Ce perfectionnement a été apporté en exécution en langage machine, comme on le verra au paragraphe X - E - 4; on réserve la valeur $\alpha = 0$ pour les boucles n'ayant pas de boucle intérieure. Pour ces boucles, en exécution "langage machine", la partie de formation des ordres est supprimée après le 1er passage. Cela permet de gagner du temps (le temps est le facteur essentiel en exécution en langage machine) mais donne plus d'encombrement au programme de bouclage. C'est pour cette raison qu'en logique extérieure, quel que soit le compteur de boucles utilisé, la phase de formation des ordres n'est jamais supprimée. En logique extérieure en effet, c'est l'encombrement qui est le facteur le plus important.

- Remarque 2 : il peut arriver que l'on sorte de la boucle sans avoir décrit le nombre H de tours. Ceci peut se présenter en particulier lorsqu'il y a un test à l'intérieur de la boucle, par exemple :

.../...



Dans ce cas, on ne passe pas par les ordres de remise à zéro du comp-
eur de boucles α et du registre d'index i. S'ils doivent être utilisés par
la suite, il ne faut pas oublier de les remettre à zéro.

- Voici le programme qui exécute effectivement le bouclage (voir l'orga-
nigramme); la plupart de ses ordres sont formés par la phase de for-
mation des ordres.

B $\emptyset$ U2	R AG	C TRH α	B $\emptyset$ U13	0 0 h	0 0	
B $\emptyset$ U13	A AG	O 01	B $\emptyset$ U4	0 ... 0 h+1		
B $\emptyset$ U4	T RG	C TRH α			0 ... 0 h+1	en CTRH α
	T DI	P 0004	B $\emptyset$ U3		0 ... 0 h+1	en P0004 = 1980
B $\emptyset$ U3	S AG	P 0005		0..0 h+1-H	0 0H	
	A NG	B $\emptyset$ U5	B $\emptyset$ U11			$h + 1 < H ?$

.../...

B ØU5	R AG	C TRi	W 0006	0 0 (CTRi)	
W 0006	A AG	P 0008	B ØU7	0 0 (CTRi)+C	si $i \neq 0$
W 0006	A AG	P 0008	B ØU8		si $i = 0$
B ØU7	T RG	C TRi		0 ... 0 (CTRi)+C	en CTRi
	T DI	P 0005	B ØU8	0 ... 0 (CTRi)+C	en P0005 = 1981
B ØU8	E DI	P 0006		0 0 B	
	T DI	M EMNN	A NABØ	0 0 B	en MEMNN = 1884
B ØU11	T RD	C TRi	B ØU9	0 0	en CTRi
B ØU9	T DI	C TRH	A NABØ	0 0	en CTRH α
A NABØ	R AG	8000			
A NG	P FDEB	D EBUT			Analyse ?

La mémoire W 0006 = 1996 a un contenu différent selon que i est ou non égal à zéro. Si $i \neq 0$, W 0006 adresse à l'ordre BØU 7 = 0012 (n° 294) qui renvoie l'index modifié (CTRi) + C dans le registre d'index n° i. Si $i = 0$, W 0006 adresse directement à BØU 8 = 0013 (n° 296), n'envoyant rien en CTRi = CTR0 = 1967. C'est ainsi que le programme évite de modifier le registre d'index n° 0.

Placés par le début de la partie interprétative, H se trouve en P 0005 = 1981, C en P 0008 = 1984 et B en P 0006 = 1982.

Analyse : Les mots 1, 2, 3 de l'analyse sont déjà pourvus au moment où débute la partie propre au bouclage. Pendant cette partie, $h + 1$ (c'est-à-dire le contenu du compteur de boucles n° α après sa modification) est envoyé en P 0004 = 1980 ; le contenu du registre

.../...

d'index n° i après sa modification est envoyé en P 0005 = 1931.

Si i = 0, rien n'est envoyé en P 0005. Au dernier passage par l'ordre de bouclage, c'est 0 0H qui est envoyé en P 0004 et P 0005.

L'instruction ANABØ = 0011 (n° 300) amène le contenu du pupitre dans l'accumulateur. S'il est positif, le programme va directement en DEBUT = 1800; s'il est négatif, le programme rejoint l'analyse normale en PFDEB = 1738 (n° 40) où se perfore la carte d'analyse.

Liste du sous-programme de bouclage :

259		P RA	1500	1999			
260		P BD	0001	0042			
261		P EN	B ØU7	0012			
262		P EN	B ØU8	0013			
263	0000	R AD	W 0003	0 0	0 0		
264		D AD	0005		00000 0000		
265		A AD	0 H00		00CTRH 0000		
266		E DI	B ØU2			60 xxxx BØU13	
267		S BF	B ØU2			60CTRH 0000	en BØU2 = 0024
268		E DI	B ØU4			21 xxxx	
269		S BF	B ØU4			21CTRH 0000	en BØU4 = 0031
270		E DI	B ØU9			24 xxxx ANABØ	
271		S BF	B ØU9			24CTRH 0000	en BØU9 = 0038
272		R AD	P PPRI	0 0	0 0 9 i		
273		S AD	0 090		0 0 i		
274		A NN	B ØU1				i ≠ 0 ?
275		A AD	0 CENO	B ØU1	00 0100 0000		
276	B ØU1	A AD	0 C00		00 0 ⁰ 00 CTRi		
277		D AG	0004	0 0	00 CTRi 0000		

.../...

278	E DI	B ØU5	0 ... 0	0	00 CTRi 0000	60 xxxx W0006	
279	S BF	B ØU5		1		60 CTRi W0006	en BØU5 = 0043
280	E DI	B ØU7				21 xxxx	
281	S BF	B ØU7				21 CTRi	en BØU7 = 0012
282	E DI	B ØU11				20 xxxx BØU9	
283	S BF	B ØU11				20 CTRi BØU9	en BØU11 = 0025
284	A AG	B ØU6	10 P0008	BØU7			
285	T RG	W 0006		BØU8	10 P0008	BØU7 BØU8	en W0006 = 1996
286	B ØU2	R AG	0000	B ØU13			
287	B ØU13	A AG	0 01	B ØU4			
288	B ØU4	T RG	0000				
289		T DI	P 0004	B ØU3			
290	B ØU3	S AG	P 0005				
291		A NG	B ØU5	B ØU11			
292	B ØU5	R AG	0000	W 0006			
293	B ØU6	A AG	P 0008	B ØU7			
294	B ØU7	T RG	0000				
295		T DI	P 0005	B ØU8			
296	B ØU8	E DI	P 0006				
297		T DI	M EMNN	A NABØ			
298	B ØU11	T RD	0000	B ØU9			
299	B ØU9	T DI	0000	A NABØ			
300	A NABØ	R AG	8000				
301		A NG	P FDEB	D EBUT			
302	0 H00	00	H 0000	0000			
303	0 090	00	0000	0090			
304	0 CEN0	00	0100	0000			
305	0 C00	00	0000	C 0000			

- La mémoire W 0003 = 1993 contient α qui a été placé par le début de la partie interprétative. De même, la mémoire PPPRI = 1750 contient PP' = 9i.

- La mémoire CH00 = 0017 (n° 302) contient la constante :
 00 H0000 0000 = 00 1940 0000. L'adresse régionale H 0000 est l'adresse 1940 du compteur de boucles n° 0. Le compteur n° α a comme adresse H 000 α = 1940 + α .

.../...

- Grâce à deux PEN, on a fixé les adresses de BØU 7 et BØU 8 de telle façon que $BØU 8 = BØU 7 + 1$: $BØU 7 = 0012$ et $BØU 8 = 0013$. Dès l'instruction n° 274, profitant du fait que i est seul dans l'accumulateur, on le teste et on prépare la formation de l'ordre à envoyer en $W 0006 = 1996$. Si $i = 0$, la constante $00 0100 0000$ contenue dans la mémoire OCENØ = 0022 (n° 304) est ajoutée dans l'accumulateur droit. Après l'ordre de décalage n° 277, l'accumulateur gauche contient : 0 01 pour $i = 0$ et 0 00 pour $i \neq 0$.

L'instruction n° 284 ajoute le mot $10 P0000 BØU 7 = 10 1904 0012$ dans l'accumulateur gauche. Elle forme donc $10 P 0000 BØU 7$ si $i \neq 0$ et $10 P0000 BØU 8$ si $i = 0$. Cet ordre est ensuite envoyé en $W 0006 = 1996$.

- La mémoire OC00 = 0023 (n° 305) contient la constante : $00 0000 C0000 = 00 0000 1967$. L'adresse régionale $C 0000 = 1967$ est l'adresse du registre d'index n° 0. Le registre d'index n° i a comme adresse $C 000i = 1967 + i$.

- Ce sous-programme a été assemblé à la suite de la partie interprétative en logique extérieure, ce qui a permis de profiter des affectations de celle-ci sans avoir à réutiliser trop de pseudo-codes.

VII - D - SOUS-PROGRAMMES PLACES A POSTE FIXE.

Deux sous-programmes ont été placés à poste fixe. Ce sont : la perforation d'une séquence de mémoires et la modification de bouclage. Ces sous-programmes ont été faits de façon différente pour la logique extérieure et l'exécution en langage machine : cela a permis d'économiser de la place en logique extérieure.

VII - D - 1 - Perforation (ordre 49 N iA jB kC)

Cet ordre provoque la perforation de cartes "A mots par carte" ($A \leq 7$).

Considérons par exemple la $p^{\text{ème}}$ carte. Elle contiendra le contenu de la mémoire $B + (p - 1) A$ et des $A - 1$ suivantes si leurs adresses ne dépassent pas C . Appelons X la première mémoire représentée sur cette carte, soit $B + (p - 1) A$. La carte portera :

en mot 1	:	00 X A
en mot 2	:	le contenu de X, soit (X)
.....		
en mot 2 + m	:	(X + m)
.....		
en mot 2 + A - 1:	:	(X + A - 1)

Si au contraire la carte considérée est la dernière et si par exemple $X + 3 = C$, la carte portera :

.../...

en mot 1 : OO X 4

en mot 2 : (X)

en mot 3 : (X + 1)

en mot 4 : (X + 2)

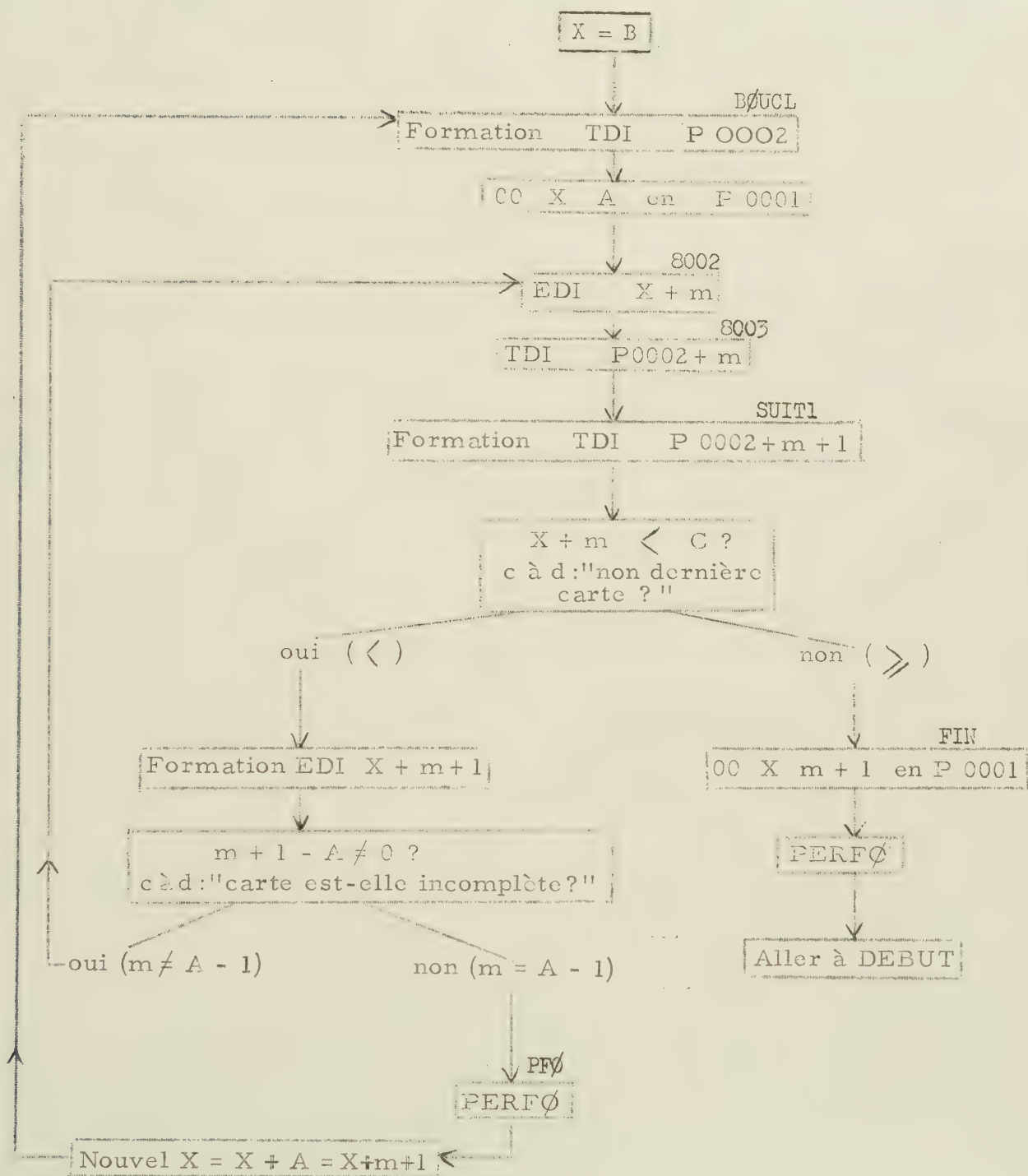
en mot 5 : (X + 3)

Pour la perforation, le mot 1 est préparé en P 0001 = 1977, le
mot 2 en P 0002 = 1978, etc

.../...

Voici l'organigramme du programme de perforation :

- 75 -



Sous-programme de perforation :

- 76 -

1	0000	E DI 6 9AG	0 0	00 A B	69 xxxx 8003	
2		D AG 0004	0 0 AA	AA B 0 ... 0		
3		S BF W 0002			69 B 8003	en W2 = 1992
4		D AD 0008	0 0	0 ... 0 A		
5		T RD W 0003			0 0 A	en W3 = 1993
6		A AG P 0000	0 ... 0 C			
7		D AG 0004	00 C 0000	00 A 0000		
8		A AD 2 4PS1		24P2 + A SUI1		
9		T RD W 0004			24 P2+A SUI1	en W4 = 1994
10		T RG W 0005			00 C 0000	en W5 = 1995
11		R AD W 0002 B ØUCL	0 0	69 B 8003		
12	B ØUCL	A AG 2 4PS1	24P0002 SUI1	69 X 8003		
13		E DI W 0003			0 0 A	
14		S BF P 0001 8002			0 X A	en P1 = 1977
	(8002)	E DI X 8003			(X)	
	(8003)	T DI P 0002 S UI1			(X)	en P2 = 1978
15	S UI1	A AG 0 10	24P2+m+1 SUI1	69 X+m 8003		
16		S AD W 0005		68 X+m-C 8003	00 C 0000	
17		E DI 8002		68 X+m-C 8003		
18		S D9 P IN				$X + m < C ?$
19		A AD 0 10		68 X+m-C+1 8003		
20		A AD W 0005		69 X+m+1 8003		
21		S AG W 0004	m + 1 - A		24 P2+A SUI1	
22		G NV P FØ				$m + 1 \neq A ?$
23		A AG 8001 8002	24P2+m+1 SUI1			
	(8002)	E DI X+m+1 8003			(X+m+1)	
	(8003)	T DI P 2+m+1 S UI1			(X+m+1)	en P2 + m+1
24	P FØ	P FØ P 0001 B ØUCL	0 0	69 X+m+1 8003		
25	F IN	R AD 8003	0 0	24 P2+m+1 SUI1		
26		S AD 2 4PS1		00 m + 1 0000		
27		E DI P 0001			00 X A	
28		D AD 0004		0 0 m + 1		
29		S BI P 0001			00 X m+1	en P1 = 1977
30		P FØ P 0001 1800				
31	6 9AG	69 0000 8003				
32	2 4PS1	24 P 0002 S UI1				
33	0 10	00 0001 0000				

- Ce programme a été écrit indépendamment de la partie interprétative et assemblé seul, dans la bande 0000. Il a ensuite été traduit. Son facteur de translation qui a été fixé à $\Delta_{49} = 1563$ est placé dans la mémoire $DEL + 49 = 1875 + 49 = 1924$. Pour l'assemblage, le programme en PASØ a été précédé des pseudo-codes suivants :

P	RA	0000	0000
P	RA	0033	1999
P	RG	P 1977	1984
P	RG	W 1991	1999

- Quand l'exécution du sous-programme débute, l'accumulateur gauche contient zéro, le droit contient 00 A B et le distributeur 20 C ANALY; en effet, l'entrée se fait comme pour un sous-programme à 3 adresses ordinaire ($PP' = 49$). Mais ce sous-programme n'est pas standard. En particulier, il ne tire pas C du distributeur mais de la mémoire P 0003 = 1984 où l'a placé la partie interprétative. Le sous-programme de perforation destiné à l'exécution en langage machine est, au contraire, standard.
- Le sous-programme utilise les mémoires de travail W (1991,) et les mémoires de perforation P (1977, ...). Or, en logique extérieure, l'analyse utilise également les mémoires P (1977, ...). A moins d'utiliser d'autres mémoires de perforation pour le programme de perforation (ce qui aurait été encombrant), il était pratiquement impossible de concilier ce programme avec l'analyse. Il a été jugé peu gênant de supprimer

.../...

l'analyse des ordres de perforation. En logique extérieure, si le signe du pupitre est -, les ordres de perforation 49 fonctionnent donc normalement mais ne sont pas analysés. Remarquons qu'au contraire les ordres de perforation simple - 06 U V sont analysés. Comme l'analyse n'existe pas pour l'ordre 49, c'est par un envoi à DEBUT = 1800 et non à ANALY = 1700 que se termine le sous-programme.

- La première partie, jusqu'à BØUCL, forme un certain nombre de mots utiles dans la suite.
- Lorsqu'on arrive à BØUCL (venant du début ou de PFØ) l'accumulateur gauche contient zéro et le droit contient 69 X 8003, X étant l'adresse de la première mémoire représentée sur la carte en préparation. Pour la 1ère carte, X = B. Pour la 2ème, X = B + A, etc... Chaque fois que l'on commence la préparation d'une carte, on revient à BØUCL. On envoie d'abord 00 X A en P 0001 = 1977 qui sera le mot 1 de la carte.
- Pour exécuter le test $X + m < C$?, on se sert d'un test SD. Dans l'accumulateur droit, se trouve 69 X + m 8003. On en soustrait 00 C 0000. Si $X + m \geq C$, on obtient 69 X + m - C 8003. Si au contraire $X + m < C$, on obtient 68 10000 + X + m - C 8003. Ainsi, en faisant un test SD sur la 9ème position de ce mot, on fait un test sur $X + m - C$. Pour cela, on amène le résultat de la soustraction dans le distributeur et on effectue le test SD9 (instruction n° 18).
- Pour obtenir $m + 1 - A$, on retranche de 24 P2 + m + 1 SUI 1 la quantité 24 P2 + A SUI 1 formée en W 0004 = 1994 par la 1ère partie.
- La branche commençant en FIN correspond à la dernière carte à perforer : cas où $X + m = C$. Dans ce cas, il faut remplacer le mot 1 par 00 X m + 1 ($1 \leq m + 1 \leq A$). Pour former m + 1, on soustrait à 24 P2 + m + 1 SUI 1 la quantité 24 P2 SUI 1 contenue dans la mémoire 24 PS 1.

VII - D - 2 - Modification de bouclage (ordre 00 N 0 1937 jB kC) :

Il arrive souvent qu'on ait à écrire un ordre de bouclage dont le nombre H de boucles soit variable. Par exemple, citons le problème suivant : on lit une carte comportant : en mot 1, x_0 ; en mot 2, h ; en mot 3, 0....0 n. Calculer $f(x_0)$, $f(x_0 + h)$, ..., $f(x_0 + n-1 h)$, f(x) étant une certaine fonction imposée. On aura, dans ce problème, à décrire n fois la boucle calculant f(x). Mais le n variera d'une carte lue à l'autre. On pourrait penser qu'il suffit d'écrire l'ordre de bouclage 9i N α 0000 O B O C et de commander l'addition en virgule fixe de la mémoire contenant 0...0 n à la lère mémoire de l'ordre de bouclage. Mais cela n'est pas possible à cause de l'autoprogrammation. En effet, en exécution en langage machine pour un ordre de bouclage placé dans les mémoires D et D + 1 par exemple, H se trouve sous la forme 0....0 H en mémoire D + 1. Ce n'est pas en mémoire D qu'il faut ajouter 0....0 n (comme en logique extérieure) mais en D + 1. On ne peut donc pas utiliser un ordre d'addition valable à la fois pour la logique extérieure et l'exécution en langage machine. Il a fallu créer un nouvel ordre, l'ordre de modification de bouclage. L'ordre de modification de bouclage 00 N 0 1937 jB kC consiste à remplacer le H de l'ordre de bouclage placé en C et C + 1 par H' placé en B sous la forme 00 0000 H'.

Le programme de modification de bouclage est différent en logique extérieure et en autoprogrammation. Il a dans les deux cas été écrit sous la forme d'un sous-programme (à 2 adresses) utilisant l'entrée standard. En logique extérieure, le facteur de translation de ce sous-programme a été fixé à $\Delta = 1556$. Comme c'est un sous-programme à 2 adresses, on aurait dû écrire l'ordre de modification de bouclage sous la forme 00 N 0 1556 jB kC. Or, comme le facteur de translation en exécution en langage

n'a pas pu être fixé à 1556, on ne peut pas appeler cet ordre 00 N 0 1556 jB kC. On a transféré les deux premières mémoires du sous-programme (1556 et 1557) en 1937 et 1938, disponibles en logique extérieure et en langage machine, et appelé l'ordre 00 N 0 1937 jB kC. Semblable opération a été effectuée pour l'exécution en langage machine.

Sous-programme de modification de bouclage :

				65 B A+1	20 C ANALY		
A =	1937	15	1558	1559	23 C ANALY	0300000000	
$\Delta +$	0002	03	0000	0000			
	3	20	1991	0004		23 C ANALY	en 1991 = W 0001
	4	69	0005	0006		69 0000 1991	
	5	69	0000	1991			
	6	22	1938	8003		69 C 1991	en 1938 = A + 1
(8003)	65	B	A+1	0.....0	0...0 H'		
+1 = (1938)	69	C	1991			91 N α H	
(1991)	23	C	ANALY			91 N α H'	en C

Les mémoires $\Delta = 1556$ et $\Delta + 1 = 1557$ n'ont plus aucune utilité et ont été libérées.

LOGIQUE EXTERIEURE

Voici la disposition sur le tambour des différentes parties de la logique extérieure:

Numéros de mémoires	Utilisations
	Sous - programmes du C.D.P. choisi
1555	
1556	
1557	Libres
1558	
1562	Modification de bouclage (début en 1937)
1563	
1595	Perforation (ordre 49)
1596	
1638	Bouclage
1639	
1670	PARTIE SPECIALE INDEXAGE
1671	
1881	PARTIE INTERPRETATIVE
1882	
1883	Libres
1884	MEMNN
1885	
1936	Mémoires réservées aux Δ_{PP} (région D)
1937	
1938	Début de modification de bouclage
1939	Libre
1940	
1949	Compteurs de boucles (région H)
1950	Libre
1951	
1960	Mémoires de lecture (région L)
1961	
1966	Libres
1967	
1976	Registres d'index (région C)
1977	
1986	Mémoires de perforation (région P)
1987	
1990	Libres mais 1987 est utilisé par le sous - programme auxiliaire de décalage
1991	
1999	Mémoires de travail (région W)

Le programme de chargement (voir paragraphe VIII-A) occupe la plupart des mémoires de 1940 à 1999. Il est donc détruit par l'exécution des calculs. Inversement, si on le réutilise après un calcul, il détruit les compteurs de boucles, les registres d'index, etc....

Les compteurs de boucles, mémoires de lecture, registres d'index, mémoires de perforation et mémoires de travail ont été placés dans la zone qu'occupe le programme de chargement parce que ces mémoires n'utilisent justement pas celui-ci, rien n'y étant enregistré. Notons que les compteurs de boucles et les registres d'index ne sont pas à zéro après le chargement (sauf le registre d'index n° 0) et qu'il est indispensable, si on les utilise, de les remettre à zéro par l'ordre - 02 U V.

Composition du paquet de cartes PASØ :

1°) Pseudocodes (n° 1 à 22)

Les pseudo-codes utilisés sont de 4 sortes : PRA, PRG, PEN, PER.

PRA : il y en a 3.

- PRA 1639 1670
 - PRA 0000 1630
- } Pourraient très bien être remplacés par

le seul ordre PRA 0000 1670

- PRA 1940 1999 En fait, cet ordre pourrait être PRA 1940 1990.

PRG et PEN : Après chaque réservation sera indiquée l'utilisation des mémoires réservées.

Les 4 premiers PRG nommés servent seulement à définir des régions, ces régions étant déjà réservées par le PRA 1940 1999.

- PRG H 1941 1949 : compteurs de boucles - H 0000 = 1940 est réservé par le PRA 1940 1999.
- PRG C 1968 1976 : registres d'index - C 0000 = 1967 est réservé par le PRA 1940 1999.
- PRG P 1977 1986 : mémoires de perforation.
- PRG W 1991 1999 : mémoires de travail.

.../...

- PRG D 1886 1939 : Mémoires réservées aux $\Delta_{PP'}$.
En fait la table de ces mémoires ne va que jusqu'à 1936. Ce PRG réserve en plus les mémoires 1937, 1938, 1939 qui pourraient aussi être réservées par une transformation du PRA 1940 1999 en PRA 1937 1999.
- PEN D 0000 1885 : Première mémoire de la table réservée aux $\Delta_{PP'}$.
Destinée à recevoir Δ_{10} .
- PRG F 1847 1855 : Premières instructions des parties interprétatives particulières aux ordres $PP' = 0P'$ (F 000 P'). cf organigramme général.
- PEN F 0000 1846 : Première instruction de la partie relative à $PP' = 00$ (sous-programmes à 2 adresses).
- PRG G 1857 1865 : Premières instructions des parties interprétatives propres à chaque P (G 000 P). cf organigramme général.
- PEN G 0000 1856 : Première instruction de la partie relative à $P = 0$ (sous-programmes à 2 adresses, opérations en virgule fixe, et tests SI et SD).
- PRG T 1867 1883 : Premières instructions des branches particulières aux PP' pour les ordres de service (T 00 PP'). Cf organigramme général, ordres à 1 mémoire.
- PEN T 0000 1866 : Première instruction de la partie relative à l'ordre
- 00 U V.
- PEN DEBUT 1800 : Première instruction de la partie interprétative.
- PEN MEMNN 1884 : Mémoire contenant l'adresse de l'instruction suivante :
00 0000 N.
- PEN ANALY 1700 : Première instruction de l'analyse.
- PEN PPPRI 1750 : Mémoire où est envoyé 0 0 PP' .
- PEN 001 1764 : Constante.
- PEN PFDEB 1738 : Ordre de perforation de la carte d'analyse.

- Ces 3 dernières réservations ont été faites pour pouvoir éviter de réassembler le sous-programme de bouclage si on réassemble le reste.

PER : PER DEL 1875 est une affectation sans réservation. En effet la 1ère mémoire utile dans la famille DEL + PP¹ est DEL + 10 = 1885 et non DEL = 1875.

2°) Partie interprétative sans index (n° 23 à 222)

- De 23 à 40: analyse
- De 41 à 47 : partie commune à tous les ordres
- De 47 à 158 : partie relative aux ordres à 2 mémoires.

Carte n° 73 : la mémoire INNIN = 1759 contient (cf VII-C-1 b) TRG

P0006 NINCH = 21 1982 1803 si le paquet "index" n'est pas chargé et

TRG P 0006 INCHA = 21 1982 1785 s'il est chargé. Dans la partie

interprétative sans index, le vrai contenu de INNIN est TRG P 0006 NINCH.

On lui a donné la valeur TRG P 0006 INCHA pour fixer INCHA dès cette

carte. Après assemblage, il a fallu remplacer 21 1982 1785 par 21 1982 1803

et ajouter au paquet d'index la carte chargeant 21 1982 1785 en INNIN = 1759.

- De 159 à 185 : constantes relatives à la partie précédente.
- De 186 à 222 : partie interprétative propre aux ordres de service.

3°) Partie "index" (224 à 258)

Il a fallu, après assemblage, ajouter au programme obtenu pour la partie

"index" la carte chargeant TRG P 0006 INCHA = 21 1982 1785 en INNIN = 1759.

4°) Sous-programme de bouclage (259 à 305)

Voir le chapitre VII - C - 4.

.../...

IX - AUTOPERFORATION - GENERALITES -

L'autoprogrammation ou compilateur provoque la formation d'un programme en langage machine équivalent au programme en CDP : pour chaque ordre en CDP est formé un groupe d'ordres en langage machine. Ce groupe sera capable de jouer le rôle de l'ordre en CDP ; certains des groupes se suffisent à eux-mêmes pour jouer ce rôle, d'autres ne sont que des entrées dans des sous-programmes.

Pour chaque ordre en CDP la machine, grâce au programme d'autoperforation, perfore une ou plus souvent 2 cartes "n mots par carte" ; ces cartes portent le morceau de programme en langage machine correspondant à l'ordre en code, ainsi que son emplacement. Pour exécuter le programme en langage machine obtenu, il faudra charger les cartes produites par l'autoperforation, jointes à un paquet vert constitué par les sous-programmes utiles. De même que pour l'instant il existe trois paquets de logique extérieure (virgule fixe, virgule flottante simple précision, virgule flottante double précision), il existe au choix trois paquets verts de sous-programmes à joindre au paquet autoperforé. Au contraire le programme d'autoperforation, qui représente la partie interprétative du procédé d'autoprogrammation, est en principe unique. En fait, comme on le verra plus loin, il faut joindre à ce paquet quelques cartes destinées à placer dans les mémoires $DEL + PP' = 1875 + PP'$ les facteurs de translation des sous-programmes constituant le paquet vert choisi. Ces cartes sont différentes pour chacun des 3 paquets verts. On a, pour plus de commodité, formé 3 paquets d'autoperforation (bleus), chacun constitué du programme interprétatif unique et des cartes chargeant les facteurs de translation voulus.

IX - A - EMPLACEMENT DU PROGRAMME EN LANGAGE MACHINE OBTENU

SEQUENCE $l_{\min}, l_{\max}$.

Le programme en langage machine est plus encombrant que celui en CDP. Il occupera, en plus de la place qu'occupait ce programme, une séquence de mémoires choisie par le programmeur. Avant l'autoperforation, celui-ci doit indiquer à la machine les limites $l_{\min}$ et $l_{\max}$ de cette séquence : il doit entrer 00 0000 $l_{\min}$ en 1832 et 00 0000 $l_{\max}$ en 1883.

Pour chaque ordre en code, le morceau de programme en langage machine correspondant occupera la place qu'occupait l'ordre (une, deux ou trois mémoires) et, si celle-ci est insuffisante, un certain nombre de mémoires de la séquence $l_{\min}, l_{\max}$. Par exemple, si le 1er ordre autoperforé exige 5 mémoires dans la séquence $l_{\min}, l_{\max}$, il occupera les mémoires $l_{\min}, l_{\min} + 1, \dots, l_{\min} + 4$. Si le 2ème ordre exige 3 mémoires de la séquence, il utilisera les mémoires $l = l_{\min} + 5, l + 1 = l_{\min} + 6, l + 2 = l_{\min} + 7$. Ainsi de suite.

Dans le morceau de programme en langage machine correspondant à un ordre en CDP placé en n (et éventuellement $n + 1$ et $n + 2$), le 1er ordre à exécuter est toujours celui qui est en n . La correspondance entre le programme initial en CDP et le programme en langage machine obtenu est ainsi évidente, ce qui facilite la consultation de celui-ci en cas de besoin et même la modification directe du programme en langage machine. Sauf pour les tests, le dernier ordre à exécuter (dans un morceau de programme correspondant à un ordre en CDP) a comme adresse instruction suivante : N .

Il se peut que la séquence $l_{\min}, l_{\max}$ choisie soit insuffisante. Dans ce cas, la machine demande la lecture d'une carte portant en mot 1 un nouveau $l_{\min}$ et en mot 2 un nouveau $l_{\max}$ (voir le paragraphe X - B).

.../...

Remarque : l'autoperforation d'un ordre ne modifie rien à cet ordre ni aux mémoires de la séquence $l_{\min}, l_{\max}$.

IX - B - NOTATIONS :

- Nous appellerons n l'adresse de l'ordre en CDP considéré. Celui-ci occupera la mémoire n, ou les mémoires n et n + 1, ou les mémoires n, n + 1 et n + 2.
- Nous appellerons l la 1ère mémoire libre de la séquence $l_{\min}, l_{\max}$ pour l'ordre en CDP considéré. Ceci suppose que les mémoires $l_{\min}$ à $l - 1$ ont été utilisées pour les ordres précédents.
- Une mémoire sert de compteur pour les l. Elle est nommée CTRL = 1332. La valeur initiale de l est $l_{\min}$ que l'on enregistre en 1332. La mémoire 1333 où est enregistré $l_{\max}$ est nommée LMAX.

IX - C - PREPARATION DES CARTES A AUTOPERFORER :

Pour la plupart des ordres en CDP, 2 cartes "n mots par carte" sont perforées. La 1ère porte les ordres qui prendront la place de l'ordre en code. Son mot 1 a la forme suivante : 00 n 000p avec $p = 1, 2$ ou 3 selon que l'ordre en code occupe une, deux ou trois mémoires. La 2ème porte les ordres à charger dans les mémoires $l, l + 1, \dots$ de la séquence $l_{\min}, l_{\max}$. Son mot 1 a la forme suivante : 00 l 000q, q étant le nombre de mémoires utilisées dans la séquence $l_{\min}, l_{\max}$ par l'ordre en C.D.P. considéré.

.../...

Le programme d'autoperforation prépare la perforation de la 1ère carte de la façon suivante :

en P 0001 = 1977	:	00 n 000p
en P 0002 = 1978	:	Mot à charger en n
en P 0003 = 1979	:	Mot à charger en n + 1 (éventuellement)
en P 0004 = 1980	:	Mot à charger en n + 2 (éventuellement)

La 2ème carte est préparée de la façon suivante :

en W 0001 = 1991	:	Mot à charger en l	} q mots
en W 0002 = 1992	:	Mot à charger en l + 1	
en W 0003 = 1993	:	Mot à charger en l + 2	
en P 0005 = 1981	:	Mot à charger en l + 3	
.....	:		
en P 0008 = 1984	:	Mot à charger en l + 6	

Après la perforation de la 1ère carte, 00 l q est envoyé en P 0001 = 1977 ; le contenu de W 0001 = 1991 est envoyé en P 0002 = 1978 ; celui de W 0002 = 1992 en P 0003 = 1979 ; celui de W 0003 = 1993 en P 0004 = 1980 . Puis la 2ème carte est perforée .

IX - D - SCHEMA GENERAL :

Comme en logique extérieure, l'interprétation d'un ordre en autoprogrammation débute toujours par l'instruction placée en DEBUT = 1800 (n° 23).

Les seules indications demandées par la machine à ce moment sont :

l'adresse n de l'ordre à interpréter (celle-ci doit être placée sous la forme 00 0000 n dans la mémoire MEMNN = 1884) et l'état du compteur CTRL = 1882 . L'autoperforation se passe suivant le schéma suivant :

.../...

- recherche de l'ordre dans sa ou ses mémoires (commence en 1800)
- découpage de l'ordre et séparation des diverses indications
- mise en place de l'adressé N de l'instruction suivante en MEMNN = 1884 .
- formation des ordres en langage machine dans les mémoires P 0002 = 1978 , P 0003 = 1979 , ... , P 0008 = 1984 et W 0001 = 1991 , W 0002 = 1992 , W 0003 = 1993 (voir le paragraphe IX - C)
- perforation de la ou des 2 cartes portant les ordres en langage machine.
- retour à DEBUT = 1800 .

Remarque : L'autoperforation s'exécute sur le programme en CDP enregistré complètement, et non carte par carte .

IX - E - ORGANIGRAMME - (voir figure page 89bis)

La première opération que fait la machine est d'amener dans l'accumulateur droit le contenu (n) de n et de tester son signe, comme en logique extérieure. Dès cet endroit se séparent les trajets suivis pour les instructions à 1 mémoire et celles à 2 mémoires . C'est en effet le signe de (n) qui les différencie. Avant d'examiner les branches relatives respectivement à (n) positif et négatif, voyons comment se passe la perforation, partie finale commune à tous les ordres.

IX - E - 1 - Perforation :

Tous les ordres à 2 ou 3 mémoires provoquent la perforation de 2 cartes sauf le t est SD non indexé) . Quand le programme parvient à l'ordre PER- $\emptyset$ = 1400 (n° 72) qui débute la partie "perforation", la 1ère carte est entièrement préparée et n'a plus qu'à être perforée . Pour la 2ème carte ,

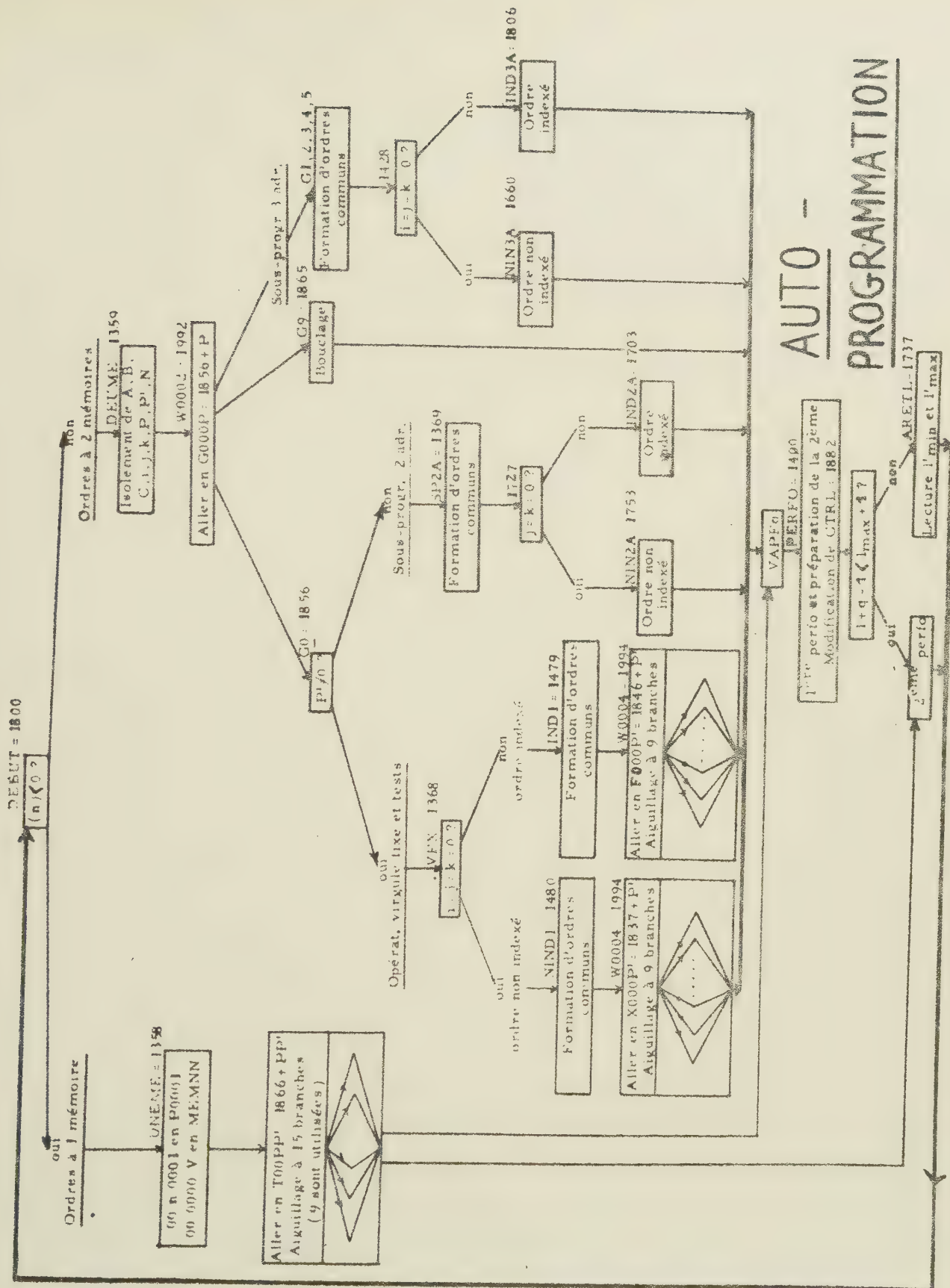
.../...

comme il est indiqué au paragraphe IX - C, la machine envoie les contenus de W 0001 = 1991, W 0002 = 1992, W 0003 = 1993 respectivement en P 0002 = 1978, P 0003 = 1979, P 0004 = 1980. Elle le fait même si q est inférieur à 3, ce qui n'est pas gênant. La machine, pour toutes les opérations citées ci-dessus, n'a besoin d'aucune indication particulière à chaque ordre en CDP. Par contre elle a besoin de connaître la valeur de q pour former 00 1 q en P 0001 = 1977 (pour la 2ème carte) et pour faire avancer le compteur CTRL = 1882. Avant d'entrer dans la partie "perforation", il faut envoyer 00 0q00 000q dans l'accumulateur droit. On a formé les constantes 00 0100 0001 en PERF 1 = 1612 (n°408), 00 0200 0002 en PERF 2 = 1662 (n° 409), 00 0500 0005 en PERF 5 = 1812 (n° 412); q ne dépasse pas 5. On a également formé les ordres : RAD PERF 1 PERFØ en VAPF 1 = 1795 (n°441), RAD PERF 2 PERFØ en VAPF 2 = 1748 (n°442),, RAD PERF 5 PERFØ en VAPF 5 = 1613 (n° 445). Toutes les branches relatives aux ordres à 2 ou 3 mémoires (sauf le test SD) rejoignent la partie "perforation" par une de ces 5 instructions.

Pour 2 ordres de service, les branches particulières se terminent par VAPF 1 = 1795 (n° 441) et, pour un autre, par VAPF 5 = 1613 (n°445). L'ordre - 05 U V, un peu particulier, sera envisagé dans l'étude détaillée. Les 5 autres ordres de service ne provoquent la perforation que d'une seule carte. Les branches relatives à ces ordres rejoignent les autres en FINPF = 1436 (n° 88).

IX - E - 2 - Instructions à 1 mémoire (cas où le contenu de n est négatif) :

- La 1ère instruction de cette branche est en UNEME = 1358 (n°462). Le début de la branche est commun à tous les ordres de service (ordres à 1 mémoire). Il comporte l'envoi de 00 n 0001 en P 0001 = 1977 (voir IX - C)



et de 00 0000 V en MEMNN = 1884. Il se termine par la préparation de l'aiguillage à branches multiples "aller à T 00 PP' = 1866 + PP' " identique à celui en logique extérieure (voir VI - C - 1).

- Les parties propres à chaque code PP' seront envisagées au cours de l'étude détaillée.
- Toutes ces parties, après la ou les perforations (voir IX - E - 1), se terminent par un retour à DEBUT = 1800 (n° 23) .

IX - E - 3 - Instructions à 2 ou 3 mémoires (cas où le contenu de n est positif) :

La 1ère instruction de cette branche est en DEUME = 1359 (n° 36).

Pour tous les ordres indexables, les cartes perforées par l'autoperforation sont différentes selon que l'ordre est indexé ou non.

a) Partie commune :

Cette partie est essentiellement formée du découpage des ordres et de la préparation du test : "aller en G 000 P = 1856 + P". Ce test est identique à celui effectué en logique extérieure (VI - C - 2 - §).

b) Branche particulière à chaque valeur de P :

α) P = 0 : la machine fait tout de suite un test sur la nullité de P'. Si P' = 0, il s'agit d'un ordre d'entrée dans un sous-programme à 2 adresses ; le programme continue en 3P2A = 1369 (n° 326). Le rôle de cette branche sera de construire et perforer les ordres nécessaires à l'entrée dans le sous-programme. Après une petite partie commune aux ordres indexés et non indexés, les branches correspondant à ces 2 cas divergent. La branche correspondant à j = k = 0 (ordre non indexé) commence à NIN2A = 1753 (n° 348) et se termine par un envoi à VAPF2. Celle qui correspond à j ou k ≠ 0 (ordre indexé) commence en IND2A = 1703 (n° 334) et se termine en VAPF3.

.../...

Si $P' \neq 0$, il s'agit d'une opération en virgule fixe ou d'un test SI ou SD ; le programme continue en VFX = 1368 (n° 133). La partie du programme d'autoperforation relative aux ordres $PP' = 0P'$ ($P' \neq 0$) est la plus encombrante. Mais l'encombrement en autoperforation est un facteur sans importance. Les branches correspondant à $i = j = k = 0$ et à i ou j ou $k \neq 0$ se séparent tout de suite. En IND1 = 1479 (n° 136) débute la branche correspondant aux ordres indexés. La machine forme un certain nombre d'instructions communes à toutes les valeurs de P' puis effectue un aiguillage suivant la valeur de P' : "aller à F000P' = 1846 + P' ". Cet aiguillage est analogue à celui de la logique extérieure sauf que le cas $P' = 0$ n'y est pas inclus. La mémoire F0009 = 1855, non utilisée pour le moment, est libre, ce qui permettra de créer éventuellement un ordre $PP' = 09$. En NIND1 = 1480 (n° 236) débute la branche correspondant aux ordres non indexés. Après une courte partie commune aux diverses valeurs de P' , s'effectue l'aiguillage "aller en X000P' = 1837 + P' ", semblable à l'aiguillage "aller à F000P' ". X0009 = 1846 est libre.

β) $P = 1$ à 5 : il s'agit des ordres d'entrée dans les sous-programmes à 3 adresses. Cette partie de programme est destinée à construire et perforer les ordres en langage machine nécessaires à l'entrée dans un sous-programme à 3 adresses. Ces ordres, comme nous le verrons plus loin, sont différents selon que l'ordre en C. D. P. est indexé ou non. Après la formation de quelques instructions communes, les branches relatives aux ordres indexés ou non se séparent. En IND3A = 1806 (n° 373) commence la branche relative aux ordres indexés et en NIN3A = 1660 (n° 392) celle relative aux ordres non indexés.

γ) $P = 8$: n'est pas encore créé.

δ) $P = 9$: ordre de bouclage. Cette partie du programme d'autoperforation forme et perfore les ordres en langage machine nécessaires à l'entrée dans le sous-programme de bouclage placé à poste fixe dans les paquets verts à joindre aux programmes autoperforés.

X - ETUDE DETAILLEE DE L'AUTOPROGRAMMATION

Quelques détails propres à certains ordres, notés dans la description de la logique extérieure, n'ont pas été répétés dans la description de l'autoprogrammation.

X - A - INDEXAGE.

Ce n'est pas pendant la phase d'autoperforation que s'exécute l'indexage. En effet, aucun ordre du programme ne s'exécute pendant cette phase, en particulier pas les ordres de bouclage. Ce n'est qu'au cours de l'exécution en langage machine du programme autoperforé que l'indexage peut se faire. Il existe pour cela 3 sous-programmes d'indexage placés à poste fixe dans les paquets verts à joindre aux programmes autoperforés. L'un de ces programmes est relatif aux opérations en virgule fixe, un autre aux sous-programmes à 2 adresses et le 3ème aux sous-programmes à 3 adresses. L'indexage des 2 ordres de tests n'utilise pas ces sous-programmes; tout leur programme d'indexage (très court) est obtenu par autoperforation (voir X - E - 2 - d - α).

Contrairement à ce qui se produit en logique extérieure, le programme d'autoperforation effectue pour chaque ordre indexable un test pour savoir s'il est indexé ou non. S'il ne l'est pas, la machine forme et perfore des instructions en langage machine destinées à exécuter l'ordre directement pour ne pas perdre de temps. S'il est indexé,

.../...

la machine forme et perfore des instructions capables d'exécuter l'ordre après la passage par l'un des 3 sous-programmes d'indexage.

Ces mots formés contiennent les indications nécessaires sous la forme la plus compacte possible.

Les 3 sous-programmes d'indexage ont été rassemblés en un seul.

Ce sous-programme unique a été écrit en langage machine dans les mémoires 0000 à 0074. Il a été ensuite perforé en cartes translatables 5 mots par carte, puis traduit. Son facteur de translation, placé en $DEL + 52 = 1875 + 62 = 1937$ pendant l'autoperforation, est égal à $\Delta_i = 1807$. Après chaque instruction du programme d'indexage nous écrivons, dans la liste qui suit, les 2 indicatifs de translation :

le 1er = 8 si l'adresse facteur doit être traduite

= 9 sinon

le 2ème = 8 si l'adresse instruction suivante doit être traduite

= 9 sinon

X - A - 1-Indexage des opérations en virgule fixe.

Prenons pour exemple l'addition. Le cas des autres opérations est tout à fait semblable, même pour les opérations avec décalages (voir

X - E - 2 - d - α).

.../...

L'autoperforation fournit les ordres suivants pour un ordre d'addition indexé 01 N iA jB kC placé en n et n + 1 :

n	RAD	n + 1	Δ_i
n + 1	60	1	$\Delta_i + 13$
1	i0	000j	000k
1 + 1	TRG	C	N
1 + 2	AAG	B	W 0002
1 + 3	RAG	A	W 0001

Ces ordres sont obtenus en langage machine pur (numérique). Comme pour tous les ordres, c'est l'instruction placée en n qui s'exécute la 1ère. Elle provoque immédiatement l'entrée dans le sous-programme d'indexage.

$\Delta_i +$	n	65	n+1	Δ_i	0 — 0	60 1 $\Delta_i + 13$	60 1 $\Delta_i + 13$	
0000	24	1993	0001	9 8			60 1 $\Delta_i + 13$	en W 0003
1	16	0002	0006	8 8		10 1+1 $\Delta_i + 25$		
2	49	9998	9988	9 9				
3	05	0001	0001	9 9				
4	04	9998	2023	9 8				
5	00	0100	0000	9 9				
6	20	1994	0007	9 8			10 1+1 $\Delta_i + 25$	en W 0004
7	15	0003	0008	8 8		15 1+2 $\Delta_i + 25$		
8	20	1995	0009	9 8			15 1+2 $\Delta_i + 26$	en W 0005
9	16	0004	0010	8 8		10 1+3 8003		
10	20	1996	1993	9 9			10 1+3 8003	en W 0006
(1993)	60	1	0013	8	i0000j000k	0 — 0		
13	30	0005	0012	9 8	00000i0000	j000k0 — 0		

.../...

Δ_i	12	10	0011	0014	8 8	60 CTRi $\Delta_i + 27$	j000k 0....0		
	11	60	1967	0027	9 8				
	14	11	8003	0015	9 8	0 ——— 0		60CTRi $\Delta_i + 27$	
	15	24	1997	0016	9 8			60CTRi $\Delta_i + 27$	en W 0007
	16	35	0001	0017	9 8	0 ——— 0 j	000k 0 ——— 0		
	17	19	0005	0018	8 8	000k 0 ——— 0	000j 0 ——— 0	0001000000	
	18	30	0002	0019	9 8	00000k 0000	00000j0000		
	19	10	0020	0021	8 8	10CTRk $\Delta_i + 24$			
	20	10	1967	0024	9 8				
	21	21	1993	0022	9 8			10CTRk $\Delta_i + 24$	en W 0003
	22	15	0023	8002	8 9		65CTRj W3		
	23	65	1967	1993	9 9				
(8002)	65	CTRj	1993			0 ——— 0	0 — 0(CTRj)		
(1993)	10	CTRk	0024	8		0 — 0(CTRk)			
	24	35	0004	1994	9 9	00(CTRk)0000	00(CTRj)0000		
(1994)	10	1+1	0025	8		21 C' N			
	25	21	1992	1995	9 9			<u>21 C' N</u>	en W 0002
(1995)	15	1+2	0026	8			10 B' W2		
	26	20	1991	1997	9 9			<u>10 B' W2</u>	en W 0001
(1997)	60	CTRi	0027	8		0 — 0(CTRi)	0 ——— 0		
	27	35	0004	1996	9 9	00(CTRi)0000			
(1996)	10	1+3	8003			<u>60 A' W1</u>			

- Cette partie du sous-programme d'indexage utilise les mémoires de travail W 0001 = 1991 à W 0007 = 1997.
- Les constantes placées en $\Delta_i + 4$, $\Delta_i + 11$, $\Delta_i + 20$ sont translatables.
- Les ordres $\Delta_i + 0$ à $\Delta_i + 10$ forment les ordres utiles par la suite pour la recherche du contenu des mémoires 1+1, 1+2 et 1+3.

.../...

- Les ordres $\Delta_i + 13$ à $\Delta_i + 22$ forment les ordres devant amener dans l'accumulateur le contenu des registres d'index CTRi, CTRj, CTRk.
- Après l'ordre placé en W 0006 = 1996, s'exécute l'addition demandée.
- L'ordre qui était à l'origine placé en 1 + 3 est exécuté en 3003 aussitôt après l'indexage.
- L'ordre qui était en 1 + 2 est, après indexage, envoyé en W 0001 = 1991 pour être exécuté.
- L'ordre qui était en 1 + 1 est, après indexage, envoyé en W 0002 = 1992 pour être exécuté.

X - A - 2 - Indexage des entrées dans les sous-programmes à 2 adresses.

L'autoperforation fournit les ordres suivants pour un ordre indexé d'entrée dans un sous-programme à 2 adresses 00 N OA jB kC placé en n et n + 1. :

n	RAG	n + 1	$\Delta_i + 23$
n + 1	60	1	$\Delta_i + 36$
1	00	000j	000k
1 + 1	RAD	B	$\Delta + 1$
1 + 2	TRD	C	N

La 1ère instruction qui s'exécute est, comme toujours, celle placée en n.

.../...

$\Delta_i +$	n	60	n+1	$\Delta_i + 28$	60	1	$\Delta_i + 36$	0.....0	60	1	$\Delta_i + 36$	
	0028	24	1991	0029	9	8			60	1	$\Delta_i + 36$	en W 0001
	29	11	0047	0030	8	8	10 1+1 W2					
	30	21	1994	0031	9	8			10	1+1	W2	en W 0004
	31	10	0046	8003	8	9	15 1+1 $\Delta_i + 32$					
(8003)	15	1+1	0032					65 B A+1				
	32	16	0045	0033	8	8		65 B A				
	33	10	0044	0034	8	8	15 1+2 $\Delta_i + 32$					
	34	69	8003	0035	9	8			15	1+2	$\Delta_i + 32$	
	35	23	1992	1991	9	9			15	1+2	A	en W 0002
(1991)	60	1	0036		8		00000j000k	0.....0				
	36	30	0004	0037	9	8	0.....0 j	000k0....0				
	37	11	8003	0038	9	8	0.....0		0.....0 j			
	38	30	0006	0039	9	8		0.....0 k				
	39	10	8001	0040	9	8	0.....0 j					
	40	35	0004	0041	9	8	00000j0000	00000k0000				
	41	10	0020	0042	8	8	10CTRj $\Delta_i + 24$		10	1967	$\Delta_i + 24$	
	42	21	1993	0043	9	8			10	CTRj	$\Delta_i + 24$	en W0003
	43	15	0023	8002	8	9		65CTRk W3	65	1967	W3	
(8002)	65	CTRk	1993				0.....0	0.....0(CTRk)				
(1993)	10	CTRj	002+		8		0.....0(CTRj)					
(24)	35	0004	1994				00(CTRj)0000	00(CTRk)0000				
(1994)	10	1+1	1992				<u>65 B' A+1</u>					
(1992)	15	1+2	<u>A</u>					<u>20 C' N</u>				
	44	00	0001	0000	9	9						
	45	00	0000	0001	9	9						
	46	04	9999	8040	9	8						
	47	49	9998	8044	9	8						

- Les constantes $\Delta_i + 46$ et $\Delta_i + 47$ sont translatables.
- Les ordres $\Delta_i + 28$ à $\Delta_i + 35$ forment les ordres qui iront chercher le contenu des mémoires $1+1$ et $1+2$.
- Les ordres $\Delta_i + 36$ à $\Delta_i + 43$ forment les ordres devant amener dans l'accumulateur le contenu des registres d'index CTRj et CTRk (A n'est pas indexable).
- L'ordre placé en W 0002 = 1992 provoque l'entrée dans le sous-programme (en A). A ce moment, les accumulateurs contiennent bien ce qui est demandé par le sous-programme.

X - A - 3 - Indexage des entrées dans les sous-programmes à 3 adresses :

L'autoperforation fournit les ordres suivants pour un ordre indexé d'entrée dans un sous-programme à 3 adresses PP' N iA jB kC placé en n et n + 1 :

n	RAD	n + 1	$\Delta_i + 40$
n + 1	6C	1	$\Delta_{PP'}$
1	iC	000j	000k
1 + 1	CC	A	B
1 + 2	2C	C	N

La 1ère instruction qui s'exécute est celle placée en n.

.../...

	n	65	n+1	$\Delta_i + 48$	0	0	60	1	$\Delta_{pp'}$		
$\Delta_i +$	0048	69	0057	0049	8	8				118003 xxxx	
	49	23	0057	0050	8	8				118003 $\Delta_{pp'}$	en $\Delta_i + 57$
	50	30	0004	0051	9	8		000060	1		
	51	35	0004	0052	9	8		60	1	0000	
	52	15	0053	0054	8	8		60	1	$\Delta_i + 63$	
	53	00	0000	0063	9	8					
	54	20	1997	0055	9	8				60	1 $\Delta_i + 63$ en W 0007
	55	16	0056	0058	8	8		15	1+1	W1	
	56	44	9998	8072	9	8					
	57	11	8003	0000	9	9					
	58	20	1994	0059	9	8				15	1+1 W1 en W 0004
	59	16	0003	0060	8	8		10	1	W0	
	60	15	0061	0026	8	8		10	1+2	$\Delta_i + 57$	
	61	00	0001	8067	9	8					
(	26)	20	1991	1997						10	1+2 $\Delta_i + 57$ en W 0001
(1997)	60	1	0063		8	i0	000j000k	0	0		
	63	30	0005	0064	9	8	00000i0000	j000k00000			
	64	10	0065	0066	8	8	15 CTRi $\Delta_i + 27$				
	65	15	1967	0027	9	8					
	66	11	8003	0067	9	8	0	0		15 CTRi $\Delta_i + 27$	
	67	24	1997	0068	9	8				15 CTRi $\Delta_i + 27$	cn W 0007
	68	35	0001	0069	9	8	00j	000k00			
	69	19	0005	0070	8	8	000k 0....0	000j0.....0			
	70	30	0002	0071	9	8	00000k0000	00000j0000			
	71	10	0072	0073	8	8	60 CTRk W7				
	72	60	1967	1997	9	9					
	73	15	0074	0062	8	8		15CTRj W4			
	74	15	1967	1994	9	9					
	62	20	1996	8003	9	9				15 CTRj W4	en W 0006
(8003)	60	CTrk	1997			00(CTrk)	0	0			
(1997)	15	CTRi	0027		8			0...0(CTRi)			
(	27)	35	0004	1996		00(CTrk)0000	00(CTRi)0000				
(1996)	15	CTRj	1994				00(CTRi)(CTRj)				
(1994)	15	1+1	1991				00 A' B'				
(1991)	10	1+2	0057		8	20 C' N					
(	57)	11	8003	$\Delta_{pp'}$		0	0	00 A' B'		20 C' N	

- Les constantes $\Delta_i + 53$, $\Delta_i + 56$, $\Delta_i + 61$, $\Delta_i + 65$ sont translatables.
- Les ordres $\Delta_i + 48$ à $\Delta_i + 60$ servent à former les ordres qui amènent dans les accumulateurs le contenu des mémoires $1 + 1$ et $1 + 2$.
- Les ordres $\Delta_i + 63$ à $\Delta_i + 74$ servent à former les ordres qui amèneront dans les accumulateurs le contenu des registres d'index CTRi, CTRj, CTRk.
- Le dernier ordre, placé en $\Delta_i + 57$, provoque l'entrée dans le sous-programme (en Δ_{PP}). A ce moment, les accumulateurs et le distributeur contiennent bien ce qu'exige le sous-programme.

X - B - PERFORATION (voir IX - E - 1)

72	P'ERFØ	P'FØ	P'0001	0 0	000q00000q	Perfo lère carte
73		A AG C	TRL	0 . . 0 1		
74		D AG	0004	00 1 000q	00000q0000	
75		T RG P	0001		00 1 q	mot 1, 2ème carte
76		A AG	8002	00 1+q 000q		
77		D AD	0004	0 . . . 01 + q	000q00000q	
78		T RG C	TRL		0 . . . 0 1+q	en CTRL = 1882
79		S AG L	MAX	0 01+q-1 _{max}		
80		S AG	0102 S UITE	0 01+q-1 _{max} ⁻²		
81	S UITE	E DI W	0001			
82		T DI P	0002		mot 2	2ème carte
83		E DI W	0002			
84		T DI P	0003		mot 3	2ème carte

85	E	DI	W	0003		$1+q-1_{\max}^2$			
86	T	DI	P	0004			mot 4	2ème carte	
87	A	NG	F	INPF	A	RETL		$1+q-1_{\max}^2 < 0 ?$	
88	F	INPF	P	FØ	P	0001	D	EBUT	
456	A	RETL	L	EC		1951			Nouveaux $1_{\min}$ et $1_{\max}$
457	E	DI		1951			0...0 $1'_{\min}$		
458	T	DI	C	TRL			0...0 $1'_{\min}$	en CTRL = 1882	
459	E	DI		1952			0...0 $1'_{\max}$		
460	T	DI	L	MAX			0...0 $1'_{\max}$	en LMAX = 1883	
461	R	AD	M	EMIN	D	EBU2	0.....0	00 n 0000	

- L'instruction n° 78 envoie en CTRL = 1882 l'adresse $1+q$ de la

1ère mémoire de la séquence $1_{\min}$, $1_{\max}$ disponible pour l'ordre suivant (la dernière mémoire utilisée pour l'ordre considéré est la mémoire $1+q-1$).

- Avant de perforer la deuxième carte, il faut vérifier si $1+q-1$ ne

dépasse pas $1_{\max}$. Pour cela, est effectué le test : $1+q-2-1_{\max} < 0 ?$

Si la réponse est OUI, cela veut dire que $1+q-1 < 1_{\max} + 1$,

c'est-à-dire que $1+q-1 \ll 1_{\max}$. La carte peut être perforée et on

peut passer à l'ordre suivant. Si la réponse est NON, cela veut dire que

$1+q-1 \gg 1_{\max} + 1$, c'est-à-dire que $1+q-1 > 1_{\max}$. La limite

supérieure $1_{\max}$ de 1 est atteinte. La 2ème carte de l'ordre n'est

pas perforée. La machine, en ARETL = 1737 (n° 456), demande à

connaître une nouvelle séquence l'_{min} , l'_{max} par la lecture d'une carte, non chargement, portant 00 0000 l'_{min} en mot 1 et 00 0000 l'_{max} en mot 2. A ce moment, l'interprétation de l'ordre considéré recommence à DEBU2 = 1356 (n° 26) (voir X-C). La 1ère carte relative à cet ordre et qui a déjà été perforée perd sa valeur et est remplacée par une autre, différente car l' est différent. Elle n'a pas besoin d'être supprimée car, lors de l'enregistrement du paquet autoperforé, ce qu'elle aura chargé en n , $n + 1$ (et éventuellement $n + 2$) sera remplacé par ce que la nouvelle carte y chargera.

- Le fait qu'il n'y ait pas zéro dans l'accumulateur droit ne change rien au test " $l + q - l_{max} - 2 < 0$?".
- La mémoire 002 = 1410 (n° 403) contient la constante 00 0000 0002.
- Pour les ordres de service qui ne provoquent la perforation que d'une carte, cette perforation est effectuée par l'ordre FINPF = 1436 (n° 33).

X - C - PARTIE COMMUNE A TOUS LES ORDRES.

Au début de cette partie, dont la 1ère instruction est en DEBUT = 1800, se trouve dans la mémoire MEMNN = 1884 la quantité 00 0000 n indiquant l'adresse n de l'ordre à autoperforer.

.../...

23	D	EBUT	R	AD	M	EMNN		0.....0	0.....0 n		
24			D	AG		0004			00 n 0000		
25			T	RD	M	EMN	D	EBU2		00 n 0000	en MEMN = 1403
26	D	EBU2	T	RG	P	0003			0.....0		en P3 = 1979
27			T	DI	P	0004			0.....0		en P4 = 1980
28			T	DI	P	0005			0.....0		en P5 = 1981
29			T	DI	P	0006			0.....0		en P6 = 1982
30			T	DI	P	0007			0.....0		en P7 = 1983
31			T	DI	P	0008			0.....0		en P8 = 1984
32			E	DI	F	F		F	G	65 xxxx FH	
33	F	F			65	0000		F	H		
34	F	G	S	BF	F	F		8001		65 n FH	en FF = 1390
	(8001)		R	AD		n		F	H	0.....0	(n)
35	F	H	A	NG	U	NE	E	D	EUME		test sur (n)

(UNE MEmoire, DEUX MEmoires)

Cette séquence de programme sert à former et à exécuter l'ordre 65 n F H qui amène le contenu de n dans l'accumulateur, puis à exécuter le test qui adresse : en UNEME = 1358 (n° 462) si l'ordre est à une mémoire et en DEUME = 1359 (n° 36) si l'ordre est à 2 mémoires. De plus cette séquence remet à zéro les mémoires P 0003 à P 0003 pour que les cartes perforées par l'autoperforation ne comportent pas trop de mots inutilement différents de zéro.

.../...

X - D - ORDRES A I MEMOIRE OU ORDRES DE SERVICE

X - D - 1 Partie commune à ces ordres : débute à UNEME = 1350 (n° 462)

					0.....0	-PP' U V		
462	U	NEME	T	RD	W	0002		-PP' U V en W2 = 1992
463			R	AD	M	EMN	00 n 0000	
464			A	AD	O	01	00 n 0001	
465			T	RD	P	0001	00 n 0001	en P1 = 1977
466			R	AV	W	0002	0.....0 PP' U V	
467			E	DI		8003	0.....0	
468			S	BI	M	MEMN	0...0 V	en MEMNN = 1884
469			E	DI		8003	0.....0	
470			S	BT	W	0001	00 U 0000	en W1 = 1991
471			D	AG		0002	0....OPP' U V 00	
472			A	AG	C	TE4 8003	0....OTOPP' 0...OT0000	Formation aiguillage
	(8003)	S	ØP			0000 T'00PP'		<u>Aiguillage</u>

- Pour tous les ordres de service, la 1ère carte perforée doit avoir pour mot 1 : 00 n 0001. La préparation de ce mot est donc dans la partie commune. 00 n 0001 est envoyé en P 0001 = 1977 et sera ainsi perforé en mot 1.

- V, dans la plupart des cas, représente l'adresse de l'instruction suivante. L'envoi de 00 0000 V en MEMNN = 1884 se fait donc aussi pendant la partie commune. Le V en MEMNN est destiné à indiquer l'adresse de la prochaine instruction à autoperforer. Mais V est

.../...

.....

aussi utilisé comme adresse instruction suivante du dernier ordre en langage machine à exécuter.

- La mémoire CTE4 = 1629 (n° 500) contient la constante 00 0000 T0000 = 00 0000 1866.

X - D - 2 - Partie propre à chaque code PP'

= Code -00 U V : sans opération, aller à V.

L'ordre qui devra s'exécuter en langage machine est :

00 U V . Cet ordre devra être en n. L'autoperforation doit donc produire une seule carte ayant :

en mot 1 : 00 n 0001 formé par la partie commune

en mot 2 : 00 U V

Les seuls ordres du programme d'autoperforation particuliers à

- 00 U V sont donc :

531	T	0000	RAV	W	0002	V	APFØ	0	...	0	00	U	V	
497	V	APFØ	TRD	P	0002	F	INPF					00	U	V
88	(F	INPF	Ø	P	0001	D	EBUT							

en P2 (puis n)

- FINPF = 1436 est le dernier ordre de la partie "perforation" (cf X - B)

- La mémoire W 0002 = 1992 contient - PP' UV, envoyé par l'ordre UNEME (n° 462).

= Code -01 U V : arrêt ; puis départ éventuel à V.

.../...

L' autoperforation doit produire une carte unique portant :

en mot 1 : 00 n 0001

en mot 2 : 01 U V

Le seul ordre particulier à - 01 U V est donc :

498 | T | 0001 | R | AV | W | 0002 | VI | APEØ | 0 0 | 01 U V |
 497 | (V | APEØ) | T | RD | P | 0002 | F | INPF | | 01 U V | en P2 (puis n)

= Code - 02 U V : initialisation

Cet ordre, comme en logique extérieure, a pour rôle de remettre à zéro les compteurs de boucles et registres d'index. Mais en plus, il régénère le 1^{er} ordre du sous - programme de bouclage. Comme nous le verrons plus loin (paragraphe X - E - 4 - c), pour $\alpha = 0$ le premier ordre du programme de bouclage ($24 \Delta_{bou} + 0055 \Delta_{bou} + 0001$) est remplacé par : $00 0000 \Delta_{bou} + 0035$. Cet ordre est régénéré à la sortie de boucle, en même temps que sont remis à zéro le compteur de boucles et le registre d'index utilisés. Mais si l'on ne termine pas le bouclage, l'ordre n'est pas régénéré. Exemple : on effectue un problème comportant un bouclage avec $\alpha = 0$. On doit, pour une cause quelconque, interrompre le déroulement de ce problème sans passer par la fin de boucle. Si l'on veut exécuter un autre problème sans recharger le C.D.P., le programme de bouclage ne fonctionnera pas normalement car son premier ordre sera détruit .

.../...

Pour éviter cet ennui, l'ordre - 02 U V, qui initialise les compteurs de boucles et registres d'index, initialise aussi la mémoire $\Delta_{bou} + 0000$ en y entrant : $24 \Delta_{bou} + 0055 \Delta_{bou} + 0001$.

Une partie du programme destiné à exécuter l'initialisation pendant l'exécution en langage machine est à poste fixe dans le paquet vert à joindre au paquet autoperforé. L'autre partie est fournie par l'auto-perforation.

La partie que nous appellerons la "partie fixe" occupe les mémoires $K = 1930$ à $K + 5 = 1935$. Elle représente le programme de remise à zéro des compteurs, et est analogue au programme d'initialisation en logique extérieure (voir VII - B - 2).

Voici la liste du programme qui se déroule pour l'initialisation en exécution en langage machine.

n	RAG	1	1+1	44 1932 V	0.....0	44 1932 V	
1	44	1932	V				
1+1	TDI	1991	1+2			44 1932 V	en W1 = 1991
1+2	RAG	1+3	1+4	$24 \Delta_{bou} + 55 \Delta_{bou} + 1$		$24 \Delta_{bou} + 55 \Delta_{bou} + 1$	
1+3	24	$\Delta_{bou} + 55$	$\Delta_{bou} + 1$				
1+4	TDI	Δ_{bou}	1930			$24 \Delta_{bou} + 55 \Delta_{bou} + 1$	en $\Delta_{bou} = 1751$
1930	RAG	1933	8003	20 1976 1931	0.....0		
8003	TRD	M	1931			0.....0	en M
1931	SAG	1934	1991	00 M-1940 0000			

.../...

(1991)	GNN	1932	V				M-1940 $\neq$ 0 ?
1932	AAG	1935	8003	20 M-1	1931		
1933	20	1976	1931				
1934	20	1940	1931				
1935	20	1939	1931				

- Les ordres n et $1 + 1$ servent à envoyer en W 0001 = 1991 l'ordre 44 1932 V qui dépend de l'ordre -- 02 U V et ne peut par conséquent pas être mis à poste fixe.
- Les ordres $1 + 2$ et $1 + 4$ servent à initialiser la mémoire $\Delta_{bou} = 1751$ en y envoyant $24 \Delta_{bou} + 55 \Delta_{bou} + 1 = 24 1806 1752$, contenu en $1 + 3$. Ces ordres ne dépendent que de la valeur de Δ_{bou} . Or Δ_{bou} a été fixé à 1751. On aurait pu mettre ces ordres dans la partie fixe. On ne l'a pas fait pour ne pas avoir trop de choses à modifier, avec risques d'oublis, si l'on devait modifier Δ_{bou} .
- Les ordres n et $1 + 2$ pourraient être des EDI mais la formation des RAG économise une mémoire.
- La seule différence (autre que des différences d'affectation d'adresses) entre la partie fixe de ce programme d'initialisation et celui de la logique extérieure (voir VII - B - 2) est que la réponse NON au test GNN envoie dans le 1er cas à V et dans le 2ème cas à DEBUT.

.../...

L'autoperforation doit fournir 2 cartes : la 1ère doit porter :

en mot 1 : 00 n 0001

en mot 2 : 60 1 1+1

La 2ème doit porter, K étant égal à 1930 :

en mot 1 : 00 1 0005

en mot 2 : 44 K+2 V

en mot 3 : 24 1991 1+2

en mot 4 : 60 1+3 1+4

en mot 5 : 24 $\Delta_{bou} + 55 \Delta_{bou} + 1$

en mot 6 : 24 Δ_{bou} K

Voici la partie du programme d'autoperforation propre à l'ordre - O2 U V :

503	T	0002	R	AG	C	TRL	0.....0 1	0.....0	0.....0 1	
504			D	AG		0004	00 1 0000			
505			A	AG		8001	00 1 1		0.....0 1	
506			A	AD		8001		0.....0 1	0.....0 1	
507			A	AG	R	G01	60 1 1+1			
508			T	RG	P	0002			60 1 1+1	en P2 (puis n)
509			A	AG	O	33	60 1+3 1+4			
510			T	RG	W	0003			60 1+3 1+4	en W3 (puis 1+2)
511			A	AD	2	4W12		24 W1 1+2		
512			T	RD	W	0002			24 1991 1+2	en W2 (puis 1+1)
513			R	AD	D	0051	0.....0	0.....0 Δ_{bou}	0.....0 Δ_{bou}	

.../...

514	DIAG	0004	00 Δ_{bou} 0000	
515	A AD	8001	00 Δ_{bou} Δ_{bou}	
516	E DI	2 40K		24 0000 K
517	S BF	P 0006		24 Δ_{bou} K en P6 (puis 1+4)
518	A AD	2 4551	24 Δ_{bou} +55 Δ_{bou} +1	
519	T RD	P 0005		24 Δ_{bou} +55 Δ_{bou} +1 en P5 (puis 1+3)
520	R AG	M EMNN	0.....0 V	0.....0
521	A AG	4 4K2	44 K+2 V	
522	T RG	W 0001 V APF5		44 K+2 V en W1 (puis 1)

- La mémoire RG01 = 1510 (n° 446) contient la constante 60 0000 0001
- La mémoire 033 = 1663 (n° 451) contient la constante 00 0003 0003
- La mémoire 24 W 12 = 1713 (n° 452) contient la constante 24 W 0001 0002 = 24 1991 0002.
- La mémoire 24 0K = 1763 (n° 453) contient la constante 24 0000 K = 24 0000 1930.
- La mémoire 24551 = 1813 (n° 454) contient la constante 24 0055 0001
- La mémoire 44 K 2 = 1464 (n° 455) contient la constante 44 K + 2 0000 = 44 1932 0000.
- La mémoire D 0051 = DEL + 61 contient le facteur de translation du sous-programme de bouclage, $\Delta_{bou} = 1751$.

.../...

= Code - 03 U V : remise à zéro de la mémoire U

Le programme fourni par l'autoprogrammation est :

n	RAG	0000	1	x.....x	0....0	
1	TRD	U	V			0.....0 en U

Il se perforera donc 2 cartes.

La partie d'autoprogrammation relative à l'ordre - 03 U V est :

524	T	0003	S	AG	8003		X.....X	U	V	00	
525			D	AD	0002		0.....0				
526			A	AD	T	D00					
527			T	RD	V	0001					
528			R	AG	C	TRL		0.....0	1	0.....0	
529			A	AG	R	G00		60	0000	1	
530	T	GP21	T	RG	P	0002					
			V	APF1				60	0000	1	en P2 (puis n)
											20 U V en W1 (puis 1)

- La mémoire TD 00 = 1630 (n° 424) contient 20 0000 0000

- La mémoire RG 00 = 1730 (n° 432) contient 60 0000 0000 .

= Code - 04 U V : entrée dans le distributeur du contenu de U et arrêt par l'ordre 01 U V.

Le programme fourni par l'autoprogrammation est :

n	EDI	U	1
1	ART	U	V

Il se perforera donc 2 cartes.

La partie du programme d'autoprogrammation relative à l'ordre - 04 U V est la suivante :

532	T	0004	R	AV	W	0002	0.....0	04	U	V				
533			S	AD	O	300		01	U	V	030.....0			
534			T	RD	W	0001					01	U	V	en W1 (puis 1)
535			A	AG		6,9	690.....0							
538			A	AG	C	TRL	69 0000 1							
539			S	AG		8003	0.....0	01	U	V	69 0000 1			
540			S	BF	P	0002					69	U	1	en P2 (puis n)

- La mémoire 0300 = 1778 (n° 541) contient la constante 03 0000 0000.

- La mémoire 69 = 1594 (n° 407) contient la constante 69 0000 0000.

= Code - 05 U V : aller en langage machine à U

L'autoprogrammation, pour cet ordre, provoque la création de l'ordre en langage machine : aller à U. Elle perfore une carte portant en mot 1 : 00 n 0001, et en mot 2 : 00 0000 U. Mais en plus elle perfore tout le programme en langage machine commençant à U et se terminant à l'instruction ayant 1800 comme adresse instruction suivante. Pour chaque ordre en langage machine, CFI (code opération, adresse Facteur, adresse Instruction) placé en J, la machine perfore une carte portant : en mot 1 : 00 J 0001

en mot 2 : C F I

Pour le dernier ordre, l'adresse I = 1800 est remplacée par V puis le programme interprétatif, après la perforation de la carte relative à cet ordre, continue en DEBUT = 1800 pour interpréter l'ordre suivant, V.

Remarque 1 : Le programme en langage machine commençant en U ne doit pas comporter de test : en effet après chaque ordre CFI, l'ordre suivant est indiqué par l'adresse I, si bien que la branche F ne serait pas traitée par l'autoperforation.

.../...

On peut au besoin remédier à cela en ajoutant au paquet autoperforé des cartes portant la partie de programme qui n'a pas été obtenue. Cependant, à moins de prendre des précautions, il faut que le programme en langage machine se termine par une seule rentrée en CDP. Autrement dit, si le programme en langage machine se divise en plusieurs branches (ce qui est déconseillé), il faut que ces branches se rejoignent avant la rentrée en CDP. Si ce n'est pas le cas, les parties en CDP qui suivent les parties en langage machine n'ayant pas été programmées ne le sont pas non plus.

Remarque 2 : Aucun ordre du programme en langage machine ne doit avoir d'adresse instruction suivante de la série 8000. En effet, comme le programme ne s'exécute pas en autoprogrammation, celle-ci, rencontrant une adresse I de la série 8000, part dans une fausse direction.

Il est en général facile d'éviter ces adresses - Exemple : la partie de programme :

0200	60	0201	0205	69 0300 0302	0 0
0201	69	0300	0302		
0205	10	0206	<u>8003</u>	69 0301 0302	
0206	00	0001	0000		

peut très bien s'écrire :

0200	60	0201	0205	69 0300 0302	
0201	69	0300	0302		
0205	10	0206	0207	69 0301 0302	
0206	00	0001	0000		
0207	<u>21</u>	<u>0210</u>	<u>0210</u>	69 0301 0302	en 210

Il faut, pour que l'autoprogrammation continue bien, que la mémoire 0210 contienne un ordre quelconque ayant comme adresse instruction suivante I = 0302.

La partie du programme d'autoprogrammation relative à l'ordre - 05 U V est la suivante :

473	T 0005	R AD W 0001	0.....0	00 U 0000	
474		D AD 0004		0.....0 U	
475		T RD W 0001 U NE53		0.....0 U	en W1 = 1991
476	U NE53	T RD P 0002		0.....0 U	en P2
477		P FØ P 0001		C' F' J	PERFO
478		R AD W 0001	0.....0	0.....0 U	
479		D AG 0004		00 J 0000	
480		A AD 0101		00 J 0001	0.....01
481		T RD P 0001		00 J 0001	en P1
482		E DI U NE51		67 xxx	
483		S BF U NE51 8001		67 J	en UNE51 = 1518
484	U NE51	67 0000			
	(8001)	R AV J	0.....0	C F I	
485		E DI 8003		0.....0	
486		S BI W 0001		0.....0I	en W1 = 1991
487		A AG 8001	0...0 I		
488		S AG E NCØD	0...0I-1800	0...0 1800	
489		G NN U NE52			I ≠ 1800 ?
490		A AG 8001 U NE53	0.....0 I	C F I	
491	U NE52	R AG 8002	C F 1800	0.....0	
492		A AD M EMNN		0....0 V	
493		S AG 8003	0.....0	C F 1800	
494		S BI P 0002 F INPF		C F V	en P2
	(F INPF)	P FØ P 0001 D EBUT			

.../...

- La première fois que le programme passe par l'ordre de perforation n° 477, la mémoire P 0001 = 1977 contient 00 n 0001, placé par la partie commune aux ordres de service. La mémoire P 0002 = 1978 contient 0 0 U. La carte perforée contient donc : en mot 1 : 00 n 0001
en mot 2 : 0 0 U

Le premier ordre exécuté en langage machine sera donc : "aller à U". Lorsqu'on passe à l'ordre n° 477 en venant de l'ordre n° 490, la carte perforée est de la forme : en mot 1 : 00 J 0001
en mot 2 : C F I

La carte relative à l'ordre en langage machine C F I = C F 1800 est perforée par l'ordre FINPF = 1436 (n° 88) qui renvoie à DEBUT = 1800. Cette carte est de la forme :
en mot 1 : 00 J 0001
en mot 2 : C F V

- La mémoire 001 = 1634 (n°422) contient la constante 00 0000 0001
- La mémoire ENCØD = 1729 (n°501) contient la constante 00 0000 DEBUT = 00 0000 1800.

= Code - 06 U V : perforation simple.

L'ordre qui devra s'exécuter en langage machine est : 71 U V. L'autoperforation doit donc fournir une carte portant :

en mot 1 : 00 n 0001
en mot 2 : 71 U V

Les seuls ordres particuliers à l'ordre - 06 U V dans le programme d'autoperforation sont :

523	T 0006	R AD	R D00	L ECPF	0 0	650 0				
96	L ECPF	A VD	W 0002	V APFØ		71	U	V	06	U V
497	V APFØ	T RD	P 0002	F INPF		71	U	V	en P2 (puis n)	

.../...

= Code - 07 U V : lecture

Cet ordre est équivalent à l'ordre 70 U V à condition que les cartes lues ne soient pas "chargement" (voir VII - B - 2). L'autoperforation fournit une carte portant :

en mot 1 : 00 n 0001

en mot 2 : 70 U V

Pour cela, la partie propre à l'ordre - 07 U V est :

495	T	0007	R	AD	6	300	L	ECPF	0.....0	630	0	
496	(L	ECPF)	A	VD	W	0002	V	APFØ		70	U	V
497	(V	APFØ)	T	RD	P	0002	F	INPF		07	U	V

70 U V en P2 (puis n)

- La mémoire 6300 = 1578 (n° 502) contient 63 0000 0000.

= Code - 09 U V : fin de séquence.

Cet ordre, en exécution en langage machine comme en logique extérieure, doit envoyer à U. Son utilité a été expliquée au chapitre VII - B - 2.

L'autoprogrammation doit donc fournir une carte portant :

en mot 1 : 00 n 0001

en mot 2 : 00 xxxx U "aller à U"

Par contre, pendant l'autoprogrammation, l'instruction suivante doit être prise en V, c'est-à-dire que lors du retour à DEBUT = 1800 c'est bien V qui doit figurer en MEMNN = 1884. Or V a déjà été placé en MEMNN par la partie commune aux ordres de service. Le seul ordre spécial à - 09 U V est donc, en autoprogrammation :

499	T	0009	D	AD	1	0006	V	APFØ	0	OT0009	U	V	0
497	(V	APFØ)	T	RD	P	0002	F	INPF	0	0	00	T	0009

00 xxxx U en P2 (puis n)

.../...

X - E - ORDRES A 2 (OU 3) MEMOIRES. -

Certains détails valables à la fois en logique extérieuré et en autoprogrammation n'ont été notés qu'au chapitre VII relatif à la logique extérieure.

X - E - I - Partie commune à ces ordres : débute à DEUME = 1359 (n°36)

Cette partie comporte le découpage des ordres ainsi que la construction et l'exécution du test "aller à 0000P = 1856 + P".

36	DEUME	E DI	8003	0.....0	PP'NNNi A	0.....0	
37		S BI C	0001			0....0 A	en C 0001 = 1968
38		S AD	8001		PP' Ni 0000		
39		D AG	0001	0.....0P	P' Ni 00000		
40		A AG C	0001	0....0G000P		0....0G0000	
41		T RG	0002			0....0G000P	en W 0002 = 1992
42		D AG C	0001	0.....0P			
43		D AG	0001	0.....0PP'	N i 000000		
44		S AG	8003	0.....0		0.....0PP'	
45		T DI F	PPRI			0.....0PP'	en PPPRI = 1360
46		D AG	0003	0....0NNN	i 0.....0		
47		T RG	MEMN			0.....0N	en MEMN = 1884
48		E AD C	0004			10.....0	en C 0004 = 1971
49		R AD	0001	0.....0	00 n 0000		
50		A AD	002		00 n 0002		
51		T PD	0001			00 n 0002	en Pl. (perfo)
52		A AD	010		00 n+1 0002		
53		F J	0000			67 xxx FL	
54	F J	S BF	0000			67 n+1 FL	en FJ = 1440
55	F K	S BF	8001	0.....0	jBBBBkCCCC		
56	(8001) F L	R AV	0003			0.....0	
57		E DI	0003			00 0000 C	en C 0003 = 1970
58		S BI C	0003		jBBBBk0000		
59		D AD	0001		jBBBBk000		
60		E DI	8003			0.....0	
61		S BF C	0002			00 B 0000	en C 0002 = 1969
62		S AD	8001		0j0000k000		
63		D AD	0003		0000j0000k		
64		E DI	0003			0.....C	
65		S BI C	0005			0.....0 k	en C 0005 = 1972
66		D AD	0001		00000j0000		
67		S BF C	0005			00 000j 000k	en C 0005 = 1972
68		R AG C	0003	0.....0C	0.....0		
69		D AG	0004	00 C 0000			
70		A AG	MEMN	00 C N			
71	(W 0002)	T RG	0006			00 C N	en C 0006 = 1973
		S GP	0000				

- Les mémoires C 0001 = 1968 à C 0006 = 1973 (registres d'index) servent ici de mémoires de travail. Les registres d'index n'ont en effet aucune utilité en autoperforation.
- La mémoire CTE 1 = 1388 (n° 402) contient la constante 00 0000 G 0000 = 00 0000 1856 qui, comme en logique extérieure, sert à former l'ordre "aller à G 000P" marquant la séparation entre les branches correspondant aux différentes valeurs de P. Cet ordre 00 0000 G 000P est mis en réserve en W 0002 = 1992 où il est envoyé par l'ordre n° 41.
- L'ordre n° 47 met en place l'adresse N de l'instruction suivante dans la mémoire MEMNN = 1884.
- Quand toutes les indications possibles ont été puisées de la partie gauche de l'instruction, on amène la partie droite, d'adresse $n + 1$, dans l'accumulateur. Pour cela on forme l'ordre $67\ n+1\ FL = 67\ n+1\ 1405$ qui amène dans l'accumulateur droit la valeur absolue de la partie droite de l'ordre. Avant de former cet ordre, le programme profite de la présence de n dans l'accumulateur pour former 00 $n\ 0002$, mot 1 de la 1ère carte à perforer si l'ordre est à 2 mémoires. 00 $n\ 0002$ est envoyé en P 0001 = 1977.
- Le programme d'autoperforation n'a jamais besoin d'avoir j et k séparément, mais sous la forme 00 000 $j\ 000\ k$. Ce mot est envoyé en C 0005 = 1972 par l'ordre n° 67.
- Ce programme a souvent besoin du mot 00 C N. Celui-ci est donc formé par la partie commune et envoyé en C 0006 = 1973.
- Cette partie commune s'achève par l'aiguillage multiple placé en W 0002 = 1992: "aller à G 000P = 1856 + P".

.../...

X - E - 2 - P = 0 : sous - programmes à 2 adresses - Opérations en virgule fixe - Tests.

Dès le début de cette partie, le programme relatif aux sous - programmes à 2 adresses se sépare des autres . Pour cela, la machine exécute un test sur la nullité de P' :

131	G'0000	R'AD	P'PPRI	0.....0	0.....0P'
132		A'NN	V'FX	S'P2A	

X - E - 2 - a) P' = 0 : sous - programmes à 2 adresses (cf VII - C - 2 - a)

La partie de programme propre à PP' = 00 commence à SP2A = 1369 (n° 326) .

Si l'ordre d'entrée dans un sous - programme à 2 adresses n'est pas indexé, le programme en langage machine qui doit être obtenu par l'autoperforation est (voir la normalisation, au paragraphe VII - C - 2 - a):

n	RAG	1	n+1	<u>65 B A+1</u>	0.....0
n+1	AAD	1+1	<u>A</u>		<u>20 C N</u>
1	65	B	A+1		
1+1	20	C	N		

Si l'ordre est au contraire indexé, le programme en langage machine qui doit être obtenu est, comme il est indiqué au paragraphe X - A - 2 :

n	RAG	n+1	$\Delta_i + 28$
n+1	60	1	$\Delta_i + 36$
1	00	000j	000k
1+1	65	B	A+1
1+2	20	C	N

Avant que se séparent les parties de programme relatives aux ordres d'entrée dans les sous-programmes à 2 adresses indexés et non indexés, se déroule une partie commune qui forme les mots 65 B A + 1 et 20 C N.

326	S P2A	R AG C 0001	0 — 0 A	0 — 0	
327		A AG C 0002	00 B A		
328		A AG R D01	65 B A+1	65 0000 0001	
329		T RG W 0001		65 B A+1	en W1 (puis 1 si j = k = 0)
330		R AD C 0006	0 — 0	00 C N	
331		A AD T D00		20 C N	
332		A AG C 0005	00 000j 000k		
333		G NN I ND2A N IN2A			index ?
334	I ND2A	E DI W 0001	00 000j 000k	20 C N	65 B A+1
335		T DI W 0002			65 B A+1 en W2 (puis 1 + 1)
336		T RG W 0001		00000j000k	en W1 (puis 1)
337		T RD W 0003		20 C N	en W3 (puis 1+2)
338		R AD C TRL	0 — 0	0 — 0 1	
339		D AG 0004		00 1 0000	
340		A AG M EMN	00 n 0000		
341		A AG R G1AG	60 n+1 8003	60 0001 8003	
342		A AD D 0052		00 1 Δ_i	0 — 0 Δ_i
343		A AD M EMX		00 1 Δ_i+28	0 — 0 28
344		E DI 8003		60 n+1 8003	
345		S BI P 0002		60 n+1 Δ_i+28	en P2 (puis n)
346		A AD R G08		60 1 Δ_i+36	6000000008
347		T RD P 0003 V APF3		60 1 Δ_i+36	en P3 (puis n+1)

... / ...

348	N	IN2A	T	RD	W	0002	0.....0	20	C	N	20	C	N	en W2 (puis 1 + 1)
349			R	AD	M	EMN	0.....0	00	n	0000				
350			D	AG		0002		n	000000					
351			A	AG	C	TRL	0.....0 1	n	0.....0	0.....0 1				
352			A	AD		8001	0.....0 1	n	00	1				
353			D	AG		0004	00	1	n	00	1	0000		
354			A	AG	R	G01	60	1	n+1		60	0000	0001	
355			T	RG	P	0002	C	ØDEC			60	1	n+1	en P2 (puis n)
356	C	ØDEC	A	AD	C	0001		00	1	A				
357			A	AD	A	D10	T	DP32	15	1+1	A	15	0001	0000
358	T	DP32	T	RD	P	0003	V	APF2			15	1+1	A	en P3 (puis n + 1)

- Le mot 65 B A + 1 est envoyé en W 0001 = 1991 par l'ordre n° 329. Pour les ordres non indexés, 65 B A + 1 doit figurer en mot 2 de la 2ème carte (pour être chargé en 1). Ce mot est donc bien placé en W 0001. Par contre pour les ordres indexés, 65 B A + 1 doit figurer en mot 3 (pour être chargé en 1 + 1). Il doit donc être envoyé en W 0002 = 1992, ce que fait l'ordre n° 335.

- Pour former l'adresse $\Delta_i + 28$, on appelle la mémoire D 0052 = DEL + 62 = 1875 + 62 = 1937 qui contient 00 0000 $\Delta_i = 00 0000$ 1807 puis la mémoire MEMX = 1659 (n°439) qui contient 00 0000 0028.

- La mémoire RD01 = 1701 (n°437) contient 65 0000 0001.

- La mémoire TDC0 = 1638 (n°424) contient 20 0000 0000.

- La mémoire RG1AG = 1460 (n° 405) contient 60 0001 8003. Cette constante sert à former le mot 60 n + 1 8003. En fait, l'adresse instruction suivante 8003 n'a aucune importance car le mot 60 n + 1 8003 subit ensuite un SBI.

- La mémoire RG03 = 1709 (n° 438) contient 60 0000 0008.

- La mémoire RG01 = 1510 (n° 446) contient 60 0000 0001.

- La mémoire AD10 = 1777 (n° 447) contient 15 0001 0000.

X - E - 2 - b - Partie commune aux ordres 0P' (P' ≠ 0)

Cette partie, qui débute en VFX = 1368 (n° 133), ne comporte que la formation du mot i0 000j 000k et de son test pour savoir si l'ordre est indexé ou non.

				0	0	0	OP'
133	VFX	AAG	C'0005	i0	000j	000k	
134		AAG	C'0004	i0	000j	000k	
135		G'NN	I'ND1	N'IND1			

X - E - 2 - c - Opérations en virgule fixe et tests, non indexés :

Cette partie débute en NIND1 = 1480 (n°236).

α - Programmes en langage machine élaborés par l'autoprogrammation :

Passons en revue les programmes en langage machine obtenus à partir de chacun des ordres 01, 02, ..., 08.

- Addition (01) :

n	RAG	A	n+1	(A)	0	0	
n + 1	AAG	B	1	(A) + (B)			
1	TRG	C	N		(A) + (B)		en C

- Soustraction (02) :

n	RAG	A	n+1	(A)	0	0	
n + 1	SAG	B	1	(A) - (B)			
1	TRG	C	N		(A) - (B)		en C

La seule différence avec l'addition est que le AAG est remplacé par SAG.

.../...

- Multiplication sans décalage (03) :

n	RAG	A	n + 1	(A)	0.....0	
n + 1	MUL	B	1 + 1		(A) × (B)	
1	TRD	C	N		(A) × (B)	en C
1 + 1	GNN	ARETM	1			

Le test placé en 1 + 1 vérifie que (A) × (B) n'a pas plus de 10 chiffres et qu'il occupe donc seulement l'accumulateur droit. Si ce n'est pas le cas, l'exécution en langage machine s'arrêtera par l'ordre 01 1333 8000 placé en ARETM = 1936. Ce dernier ordre figure à poste fixe dans les paquets verts à joindre aux cartes obtenues par autoperforation.

- Multiplication avec décalage (05) :

n	RAG	A	n + 1	(A)	0.....0	
n + 1	MUL	B	<u>n + 2</u>		(A) × (B)	
n + 2	<u>3x</u>	<u>000m</u>	<u>1 + 1</u>			Décalage sur (A) × (B)
1	TRD	C	N		(A) × (B)	en C
1 + 1	GNN	ARETM	1			

Les mots formés sont identiques à ceux relatifs à l'ordre 03 à l'exception des parties soulignées. La différence est donc légère. La 1ère carte perforée comportera 3 mots en plus du mot 1 qui sera 00 n 0003. En effet, l'ordre 05 occupe 3 mémoires, n, n + 1 et n + 2. Remarquons que, lors de l'exécution de ces ordres en langage machine, le décalage s'effectuera sur le produit, c'est-à-dire après l'exécution de la multiplication.

- Division sans décalage (04) :

n	RAD	A	n + 1	0.....0	(A)	
n + 1	DRR	B	1	0.....0	(A) : (B)	
1	TRD	C	N		(A) : (B)	en C

.../...

- Division avec décalage (O6) :

N	RAD	A	<u>n + 2</u>	0 0	(A)	
n + 1	DRR	B	1		(A) : (B)	
n + 2	3 x	000m	<u>n + 1</u>			Décalage sur (A)
1	TRD	C	N		(A):(B)	en C

Les mots formés sont identiques à ceux relatifs à l'ordre O4 à l'exception des parties soulignées. Comme pour l'ordre 05, la 1ère carte comportera 3 mots en plus du mot 1 qui sera 00 n 0003. Lors de l'exécution de ces ordres en langage machine, le décalage s'effectuera sur le dividende, c'est-à-dire avant la division.

- Test SI (07) :

n	RAG	C	<u>n + 1</u>	(C)	0 0	
n + 1	ANG	N	1			(C) < 0 ?
1	GNN	B	A			(C) ≠ 0 ?

Si (C) est négatif le programme en langage machine ira en N. Si (C) est positif ou nul, un deuxième test sera fait sur lui : si (C) est différent de zéro, c'est-à-dire positif, le programme ira en B ; si (C) est nul, le programme ira en A. C peut très bien désigner le pupitre (3000).

Remarque importante : en autoprogrammation, c'est N qui (comme pour les autres ordres) désigne l'instruction suivante. Pour obtenir l'autoperforation des branches commençant en A et B il faut en général mettre un ordre de fin de séquence à la fin de la branche N et un autre à la fin de la branche A ou B.

.../...

- Test SD (08) .

n	EDI	C	n + 1	(C)
+ 1	SDB	N	A	(Position B) = 8 ou 9 ?

Si la position n° B du contenu de C (C peut très bien désigner le pupitre : C = 8000) est un 8, le programme en langage machine ira en N. Si cette position est un 9, le programme ira en A. Si cette position n'est ni 8 ni 9, le programme s'arrêtera sur l'ordre SDB N A = 9B N A.

Remarque importante : Comme pour l'ordre 07, lors de l'autoprogrammation c'est l'adresse N qui désigne l'instruction suivante. Pour obtenir l'autoperforation de la branche A, il faut en général placer un ordre de fin de séquence à la fin de la branche N.

- Remarque : l'ordre de succession des différentes instructions en langage machine formées a été choisi pour que le programme d'autoprogrammation soit le plus simple possible et que ses différentes branches aient le plus possible de parties communes.

β) Partie commune aux ordres OP' non indexés.

Cette partie qui débute à NIND 1 = 1480 (n° 236) comporte la formation de l'ordre d'aiguillage "aller à XOOOP' = 1837 + P' " et des mots 00 A n + 1 et 00 B 00 utiles pour la plupart des branches.

.../...

236	N IND1	A AD	C TE3	0.....0	0.....0 P'	0 ... 0X000P'	0 ... 0X0000	
237		T RD	W 0004			0 ... 0X000P'	0 ... 0X0000	en W4 = 1994
238		R AG	C 0001	00AAAA	0 0			
239		D AD	0002	00AA	AA 0... 0			
240		A AD	M EMN	00AA	AA n 0000			
241		D AG	0006	00 A n	00			
242		A AG	0101	00 A n+1			0 01	
243		A AD	C 0002 W 0004		00 B 0000			
	(W 0004)	S ØP	0000 X 000P'					Aiguillage

- La mémoire CTE3 = 1833 (n° 414) contient la constante 00 0000 X0000 = 00 0000 1837.

- La mémoire 001 = 1634 (n° 422) contient 00 0000 0001.

8) P' = 1 : addition en virgule fixe (sans index)

Cette partie débute par le seul ordre qui soit différent de ceux de la partie "soustraction", c'est-à-dire la formation de 10 B 0000.

Nous verrons que la branche "soustraction" commence par la formation de 11 B 0000 puis rejoint la branche "addition".

244	X'0001	A AD	A G00	A DSØ2	00 A n+1	00 B 0000	10 0	
246	A DSØ2	A AD	C TRL			10 B 1		
247		T RD	P 0003				10 B 1	en P3 (puis n+1)
248		A AG	R G00		60 A n + 1		60 ... 0	
249		T RG	P 0002				60 A n+1	en P2 (puis n)
250		R AG	T G00	A SDIV	210 0	0 0		
251	A SDIV	A AG	C 0006		21 C N		00 C N	
252		T RG	W 0001	V APF1			21 C N	en W1 (puis 1)

.../...

- La mémoire AG00 = 1492 (n° 430) contient 10 0000 0000
- La mémoire RG00 = 1738 (n° 432) contient 60 0000 0000
- La mémoire TG00 = 1588 (n° 419) contient 21 0000 0000

§) P' = 2 : soustraction en virgule fixe (sans index)

Cette partie est identique à la précédente, sauf que c'est 11 B 1 qui doit être envoyé en P 0003.

Le seul ordre particulier à la soustraction est donc :

245	X 0002	A AD	S G00	A DSØ2	00 A n+1	00 B 0000	11 B 0000	110 0
-----	--------	------	-------	--------	----------	-----------	-----------	-------------

- En ADSØ2 = 1797 (n° 246) la branche "soustraction" rejoint celle d'addition
- La mémoire SG00 = 1542 (n° 431) contient 11 0000 0000

ε) P' = 3 : multiplication sans décalage (sans index)

253	X 0003	A AG	R G00		00 A n+1	00 B 0000		
254		T RG	P 0002		60 A n+1		60 0	
255		R AG	8002		00 B 0000	0 0		
256		A AG	M ULOZ		19 B Z		19 0000 Z	
257		A AD	C TRL			0 01		
258		A AD	0 01			0...0 1+1		
259		E DI	8003	R EUN3			19 B Z	
260	R EUN3	S BI	P 0003				19 B 1+1	en P3 (puis n+1)
261		R AG	C TRL		0 ... 01	0 0		
262		A AG	G NARM		44 ARETM 1		44 ARETM 0000	
263		T RG	W 0002				44 ARETM 1	en W2 (puis 1+1)
264		A AD	C 0006				00 C N	

.../...

265		A AD T D00			20 C N 20 0
266		T RD W 0001 VIAPF2			20 C N en W1 (puis 1)

- Cette partie débute par l'utilisation de 00 A n+1 placé dans l'accumulateur gauche, puis prépare l'ordre qui sera différent de celui formé pour la multiplication avec décalage.

En REUN3, comme nous le verrons plus loin, les parties relatives aux ordres 03 et 05 se rejoignent.

- La mémoire MULOZ = 1402 (n° 421) contient la constante 19 0000 Z = 19 0000 1939. Cette constante a été formée pour la multiplication avec index, comme nous le verrons au paragraphe X - E - 2 - d - f -. Dans le cas présent, l'adresse "instruction suivante" est ignorée.

- La mémoire GNARM = 1598 (n° 434) contient 44 ARETM 0000 = 44 1936 0000 (voir X - E - 2 - C - α).

7) P' = 5 : multiplication avec décalage (sans index).

267	X 0005	A AG R G00		00 A n+1 00 B 0000		
268		T RG P 0002		60 A n+1		60 A n+1 en P2 (puis n)
269		R AG 18002		00 B 0000 0 0		
270		A AD P 0001			00 n 0002	
271		A AD 0 01			00 n 0003 0 01	
272		T RD P 0001			00 n 0003 en P1 (mot 1)	

.../...

273		A AD	0 20		00 B 0000	00 n+2 0003	00 0002 0000	
274		A AD	C TRL			00 n+2 1+3		
275		S AD	0 02			00 n+2 1+1	0 02	
276		E DI	E D2GC				69 xxxx GC	
277		S BF	E D2GC	8001			69 n+2 GC	en ED2GC = 1419
	(8001)	E DI	n+2	G C			3x 000m Z	(Z = 1939)
278	G C	S BI	P 0004				3x 000m 1+1	en P4 (puis n+2)
279		A AG	M ULOZ		19 B Z			
280		E DI	8003				19 B Z	
281		D AD	0004	R EUN3	0000 x x 00 n+2		19 B Z	
260	(R EUN3)	S BI	P 0003	:			19 B n+2	en P3 (puis n+1)
	:	:	:	:				
	:	:	:	:				
	:	:	:	:				

- Cette branche débute par l'utilisation de 00 A n+1. Puis le mot 1 de la 1ère carte à perforer, qui est 00 n 0003 et non plus 00 n 0002, est envoyé en P 0001 = 1977. Ensuite le mot 00 n+2 1 + 1 est formé dans l'accumulateur droit ; il servira à construire 3 instructions. La 1ère de ces instructions est celle qui va chercher l'ordre de décalage contenu en n + 2. Cet ordre a la forme : 3x 000m 1939. Par un SBI, on le transforme en 3x 000m 1+1 et on l'envoie en P0004 = 1980 pour qu'il se perfore en mot 4 de la 1ère carte. Lorsque le programme arrive à REUN3 = 1548 (n° 260), l'instruction 19 B est prête dans le distributeur et n + 2 est en position d'adresse instruction suivante dans l'accumulateur droit. L'ordre REUN3 forme ainsi l'ordre 19 B n+2 et l'envoie en P0003 = 1979. Le programme se poursuit comme pour la multiplication sans index.

.../...

- Les ordres n° 273 et 275 pourraient être réunis en un seul qui ajouterait la constante 00 0001 9998. Mais l'économie d'une ou deux mémoires n'a aucune importance en autoprogrammation.
- La mémoire ED2GC = 1419 (n° 433) contient la constante 69 0002 GC = 69 0002 1700. En fait l'adresse facteur est modifiée mais n'a aucune importance.

θ) P' = 4 : Division sans décalage (sans index) :

282	X 0004	A AD	D R00		00 A n+1	00 B 0000		
						64 B 0000	640	 0
283		A AD	C TRL			64 B 1		
284		T RD	P 0003				64 B 1	en P3 (puis n+1)
285		A AG	R D00		65 A n+1		650	 0
286		T RG	P 0002	R EUN4			65 A n + 1	en P2 (puis n)
287	R EUN4	R AG	T D00	A SDIV	20 0			
251	(A SDIV)	A AG	C 0006		20 C N		00 C N	
252	()	T RG	W 0001	V APF1			20 C N	en W1 (puis 1)

- Le programme rejoint les branches relatives à l'addition et à la soustraction en ASDIV = 1745 (n° 251).

λ) P' = 6 : Division avec décalage (sans index) :

288	X 0006	A AD	D R00		00 A n+1	00 B 0000		
						64 B 0000	640	 0
289		A AD	C TRL			64 B 1		
290		S AD	8002			0 0	64 B 1	
291		T DI	P 0003				64 B 1	en P3 (puis n+1)
292		A AD	P 0001			00 n 0002		
293		A AD	0 01			00 n 0003	0.....0 1	

... / ...

294	T RD	P 0001		00 A n+1	00 n 0003	00 n 0003	en P1 (mot 1)
295	A AD	O 20			00 n+2 0003	00 0002 0000	
296	E DI	E DOGD				69 xxxx GD	
297	S BF	E DOGD				69 n+2 GD	en EDOGD = 1354
298	R AD	8003	E DOGD	0 0	00 A n+1		
	(EDOGD)	E DI	n + 2	G D		3x 000m 1991	
299	GD	S BI	P 0004			3x 000m n+1	en P4 (puis n+2)
300	A AD	R D01			65 A n+2	65 0000 0001	
301	T RD	P 0002	R EUN4			65 A n+2	en P2 (puis n)
287	(REUN4)	R AG	T D00	A SDIV	20.... 0	0.....0	
251	(ASDIV)	A AG	C 0006		20 C N	00 C N	
252	()	T RG	W 0001	V APF1		20 C N	en W1 (puis 1)

- L'ordre n° 294 envoie en P0001 le nouveau mot 1 de la lère carte à perforer, c'est-à-dire 00 n 0003 et non plus 00 n 0002.
- La mémoire ED0GD = 1354 (n° 436) contient 69 0000 GD = 69 0000 1651. En fait l'adresse facteur est modifiée mais n'a aucune importance. L'ordre 69 n+2 GD envoyé dans cette mémoire par l'ordre n° 297 sert à amener dans le distributeur l'ordre de décalage, qui est en n+2.
- La mémoire RD01 = 1701 (n° 437) contient 65 0000 0001.
- Pour la formation de 20 C N, cette branche rejoint celle relative à la division sans décalage à l'instruction REUN4 = 1686 (n° 287).

m) P' = 7 : Test SI (non indexé) :

302	X 0007	A AD	C 0001		00 A n+1	00 B 0000	
303		A AD	4 400			00 B A	
304		T RD	W 0001			44 B A	440 0
						44 B A	en W1 (puis 1)

.../...

305	R AD	M EMN	0	0	00 n 0000	
306	D AG	'0002			n 00 0000	
307	A AG	C'0003	0	0 C	n 00 0000	
308	A AD	M EMN	0	0 C	n 00 N	
309	D AG	'0004	00 C	n	00 N 0000	
310	A AG	R G01	60 C	n+1		60 0000 0001
311	A AD	C'TRL			00 N 1	
312	A AD	4,600			46 N 1	46 0000 0000
313	T RD	P 0003	T GP21		46 N 1	en P3 (puis n+1)
(T GP21)	T RG	P'0002	V APF1		60 C n+1	en P2 (puis n)

- Cette partie n'utilise pas le mot 00 A n+1 .

- La mémoire 4400 = 1476 (n° 208) contient 44 0000 0000.

- La mémoire RG01 = 1510 (n° 446) contient 60 0000 0001.

- La mémoire 4600 = 1655 (n° 314) contient 46 0000 0000.

- L'ordre TGP21 = 1552 (n° 530) a été créé pour la partie relative à l'ordre de service - 03 U V.

✓) P' = 8 : test SD (non indexé)

Cet ordre est le seul ordre à 2 mémoires qui ne provoque la perforation que d'une seule carte. La partie d'autoprogrammation qui lui est relative se termine par l'envoi à VAPFØ = 1759 (n° 497) qui dirige vers FINPF = 1436 (n° 88) sans passer par PERFØ = 1400 (n° 72) : voir l'organigramme général.

.../...

315	X 0008	A AD	M EMN		00 A n+1	00 B 0000		B = 000B
316		D AG	0004		xx n+1 0000	0B N 0000		
317		A AD	C 0001			0B N A		
318		A AD	9 00			9B N A	90 0	
319		T RD	P 0003				9B N A	en P3 (puis n+1)
320		R AD	M EMN		0 0	00 n 0000		
321		D AG	0002			n 00 0000		
322		A AG	C 0003		00 0000 C	n 00 0000		
323		D AD	0006			00 C n		
324		A AD		V APFØ		69 C n+1	69 0000 0001	
325		69	0000	0001				
497	V APFØ	T RD	P 0002	F INPF		69 C n+1		en P2 (puis n)

- La mémoire "900" = 1526 (n° 218) contient 90.....0.

X - E - 2 - d -) Opérations en virgule fixe et tests, indexés.

Cette partie débute en IND1 = 1479 (n° 136).

α) Programmes en langage machine élaborés par l'autoprogrammation.

Ces programmes, à l'exception de ceux relatifs aux ordres 07 et 08, constituent tous des entrées dans le sous-programme d'indexage commençant à $\Delta_i = 1807$ (voir X - A - 1). Ils comportent tous :

en n : RAD n+1 Δ_i
 en n+1 : 60 1 $\Delta_i + 13$
 en l : io 000j 000k

en l+1 : l'ordre à indexer par k
 en l+2 : l'ordre à indexer par j
 en l+3 : l'ordre à indexer par i

.../...

A la sortie du sous-programme d'indexage, c'est l'ordre qui était placé en 1 + 3 qui, indexé, sera exécuté le 1er ; l'ordre placé en 1 + 2, indexé, se trouve alors en W 0001 = 1991 et l'ordre en 1 + 1 se trouve alors en W 0002 = 1992.

- Addition (01) :

n	RAD	n+1	Δ_i	
n+1	60	1	$\Delta_i + 13$	
1	i0	000j	000k	
1+1	TRG	C	N	ira en W0002
1+2	AAG	B	W0002	ira en W0001
1+3	RAG	A	W0001	

- Soustraction (02) :

Ordres communs en n, n+1, 1				
1+1	TRG	C	N	ira en W0002
1+2	SAG	B	W0002	ira en W0001
1+3	RAG	A	W0001	

- Multiplication sans décalage (03) :

Ordres communs en n, n+1, 1				
1+1	TRD	C	N	ira en W0002
1+2	MUL	B	Z	ira en W0001
1+3	RAG	A	W0001	

... ..

... / ...

En Z = 1939 se trouve à poste fixe pour l'exécution en langage machine l'ordre de test : GNN ARETM W 0002 = 44 1936 1992. L'ordre des instructions en langage machine exécutées à la sortie du sous-programme d'indexage est donc :

	RAG	A'	W0001
W0001	MUL	B'	Z
Z	GNN	ARETM	W0002
W0002	TRD	C'	N

- Multiplication avec décalage (05) :

Ordres communs en 'n, n+1, 1				
1+1	TRD	C	N	ira en W0002
1+2	MUL	B	<u>n + 2</u>	ira en W0001
1+3	RAG	A	W0001	
n+2	3 x	000m	Z	Z = 1939

Les mots formés sont identiques à ceux relatifs à l'ordre 03 à l'exception des parties soulignées.

L'ordre d'exécution des instructions à la sortie du sous-programme d'indexage sera :

	RAG	A'	W0001	
W0001	MUL	B'	n + 2	
n + 2	3x	000m	Z	Décalage sur le produit
Z	GNN	ARETM	W0002	Z = 1939
W0002	TRD	C'	N	

... / ...

- Division sans décalage (04) :

Ordres communs en n, n+1, 1				
1+1	TRD	C	N	ira en W0002
1+2	DRR	B	W0002	ira en W0001
1+3	RAD	A	W0001	

- Division avec décalage (06) :

Ordres communs en n, n+1, 1				
1+1	TRD	C	N	ira en W0002
1+2	DRR	B	W0002	ira en W0001
1+3	RAD	A	<u>n + 2</u>	
n+2	3 x	000m	W0001	W0001 = 1991

Les mots formés sont identiques à ceux relatifs à l'ordre 04, à l'exception des parties soulignées.

L'ordre d'exécution des instructions à la sortie du sous-programme d'indexage sera :

	RAD	A'	<u>n + 2</u>	
n+2	3 x	000m	W0001	décalage sur le dividende
W0001	DRR	B'	W0002	
W0002	TRD	C'	N	

- Test SI (07) :

L'indexage du test SI ne se fait pas par le sous-programme d'indexage. Pour cet ordre en effet, les adresses A, B et C, dans les instructions en langage machine à exécuter, ont une disposition qui n'est compatible avec aucune des 3 parties du sous-programme d'indexage existant. Il faudrait donc probablement créer une 4ème partie pour le sous-programme d'indexage.

.../...

Il a été décidé que, pour simplifier l'indexage, seule l'adresse C serait indexable ; ainsi le programme d'indexage de l'ordre 07 est assez simple pour être obtenu par autoperforation. Les cartes perforées sont :

n	RAG	CTRk	n+1	0 ... 0 (CTRk)	0 0
n + 1	DAG	0004	1	00 (CTRk) 0000	
1	AAG	1 + 1	8003	60 C' 1 + 2	60 C 1+2
1 + 1	60	C	1 + 2		
(8003)	RAG	C'	1 + 2	(C')	0 0
1 + 2	ANG	N	1 + 3		
1 + 3	GNN	B	A		

Voir la remarque relative au test SI au paragraphe X - E - 2 - C - α -

- Test SD (08) :

Le cas de ce test est analogue à celui du test SI. C'est également l'autoprogrammation qui fournit le programme d'indexage, simplifié par le fait que seul C est indexable.

Les cartes obtenues par l'autoperforation sont :

n	RAG	CTRk	n + 1	0 ... 0 (CTRk)	0 0
n + 1	DAG	0004	1	00 (CTRk) 0000	
1	AAG	1 + 1	8003	69 C' 1 + 2	
1 + 1	69	C	1 + 2		
(8003)	EDI	C'	1 + 2		(C')
1 + 2	SDB	N	A		

Voir la remarque relative au test SD au paragraphe X - E - 2 - C - α -

3) Partie commune aux ordres 0P' indexés :

Cette partie débute à l'instruction IND1 = 1479 (n° 136). Elle est en grande partie inutile pour les ordres 07 et 08, mais n'est pas nuisible car elle n'allonge qu'extrêmement peu, ou pas, l'autoprogrammation de ces ordres. Elle forme les 3 instructions communes aux ordres 01, 02, ... 06. De plus, elle construit l'aiguillage "aller à F000P' = 1846 + P'".

136	IND1	A AD	C TE2	i0000j000k	0 OP'	0 ... 0 F000P'	0 ... 0 F0000	
137		T RD	W 0004				0 ... 0 F000P'	en W4 = 1994
138		T RG	W 0001				i0000j000k	en W1 (puis 1)
139		R AD	C TRL	0 0	0 01			
140		D AG	0004		00 1 0000			
141		A AD	D 0052		00 1 Δ_i		0 0 Δ_i	
142		A AG	8001	0 ... 0 Δ_i				
143		A AG	M EMN	00 n Δ_i				
144		A AG	R D10	65 n+1 Δ_i			65 0001 0000	
145		T RG	P 0002				65 n+1 Δ_i	en P2 (puis n)
146		A AD	6 013		60 1 $\Delta_i + 13$		60 0000 0013	
147		T RD	P 0003				60 1 $\Delta_i + 13$	en P3 (puis n+1)
148		R AD	C 0002	W 0004	0 0	00 B 0000		
	(W 0004)	S ØP	0000	F 000P'				

- C'est en D0052 = DEL + 62 = 1875 + 62 = 1937 que se trouve, pendant l'autoprogrammation, le facteur de translation du sous-programme d'indexage :

$$\Delta_i = 1807 \text{ (voir X - } \Delta \text{)}.$$

- La mémoire CTE2 = 1532 (n° 413) contient 00 0000 F 0000 = 00 0000 1846.

- La mémoire RD10 = 1560 (n° 415) contient 65 0001 0000.

- La mémoire 6013 = 1484 (n° 416) contient 60 0000 0013.

.../...

8) P' = 1 : Addition en virgule fixe (avec index) :

Cette partie débute par le seul ordre qui soit différent de ceux de la partie "soustraction", c'est-à-dire la formation de 10 B W0002.

149	F0001	AAD	AGOW2	ADSØ1	0 0	00 B 0000	10 0000 W2	
151	ADSØ1	TRD	W0003			10 B W2	10 B W2	en W3 (puis 1 + 2)
152		RAD	C0001		0 0	0 0A		
153		DAG	0004			00 A 0000		
154		AAD	RGOW1			60 A W1	60 0000 W1	
155		TRD	P0005			60 A W1	60 A W1	en P5 (puis 1 + 3)
156		AG	TGOO	ASMUL	210 ... 0		210 0	
157	ASMUL	AG	C0006		21 C N		00 C N	
158		TRC	W0002	VAPF4			21 C N	en W2 (puis 1 + 1)

- La mémoire AGOW2 = 1450 (n° 417) contient 10 0000 W0002 = 10 0000 1992.
- La mémoire RGOW1 = 1536 (n° 420) contient 60 0000 W0001 = 60 0000 1991.
- La mémoire TGOO = 1588 (n° 419) contient 21 0000 0000.

8) P' = 2 : Soustraction en virgule fixe (avec index) :

Cette partie est identique à la précédente, à part la 1ère instruction.

La seule instruction particulière est :

150	F0002	AAD	SGOW2	ADSØ1	0 0	00 B 0000	11 B W2	11 0000 W2
-----	-------	-----	-------	-------	-----------	-----------	---------	------------

- C'est 11 B W0002 au lieu de 10 B W0002 qui est envoyé en W0003 = 1993 par l'ordre ADSØ1 = 1455 (n° 151).
- La mémoire SGOW2 = 1401 (n° 418) contient 11 0000 W0002 = 11 0000 1992.

.../...

ε) P' = 3 : multiplication sans décalage (avec index)

159	F 0003	A AD	M ULOZ		0 0	00 B 0000	19 0000 Z	
160		T RD	W 0003	R EUN1		19 B Z	19 B Z	en W 3 (puis 1 + 2)
161	R EUN1	R AG	C 0001		0 0 A	0 0		
162		D AG	0004		00 A 0000			
163		A AG	R GCW1	M UD11	60 A W 1		60 0000 W1	
164	M UD11	T RG	P 0005				60 A W1	en P 5 (puis 1 + 3)
165		R AG	T DOO	A SMU1	20 0	0 0		
157	(A SMU1)	A AG	C 0006		20 C N		00 C N	
158	()	T RG	W 0002	V APF4			20 C N	en W 2 (puis 1 + 1)

- Cette partie commence par la formation de l'instruction 19 B Z = 19 B 1939 qui n'est pas utile pour l'ordre de multiplication avec décalage. A partir de l'instruction REUN 1 = 1596 (n° 161), le programme est valable à la fois pour les 2 ordres de multiplication, 03 et 05. A partir de MUDI 1 = 1691 (n° 164), le programme est valable à la fois pour les 2 ordres de multiplication et les 2 ordres de division. A partir de ASMU 1 = 1593 (n° 157), ce programme rejoint celui relatif à l'addition et la soustraction. Ces recoupements sont aisés parce que tous les ordres d'opérations en virgule fixe indexés provoquent la perforation d'une 2ème carte ayant le même nombre q = 4 de mots en plus du mot 1 (ces mots se chargeront en 1, 1 + 1, 1 + 2, 1 + 3).

- La mémoire MULOZ = 1402 (n° 421) contient 19 0000 Z = 19 0000 1939.
(voir X - E - 2 - d - α)

.../...

7) P' = 5 : multiplication avec décalage (avec index)

				0..... 0	00 B 0000		
166	F 0005	A AD	M ULOZ		19 B Z	19 0000 Z	
167		T RD	W 0003			19 B Z	en W3
168		R AD	P 0001		00 n 0002		
169		A AD	O 01		00 n 0003	0..... 01	
170		T RD	P 0001			00 n 0003	en P1 (mot 1)
171		D AD	0004		0..... On		
172		A AD	O 02		0 .. On+2	0 02	
173		E DI	W 0003			19 B Z	
174		S BI	W 0003			19 B n+2	en W3(puis 1+2)
175		D AG	0004		00 n+2 0000		
176		A AD	E DOGA	18002	69 n+2 GA	69 0000 GA	
	(8002)	E DI	n+2	G A		3x 000m Z	Z = 1939
177	G A	T DI	P 0004	R EUN1		3x 000m Z	en P4 (puis n+2)

- Ce programme commence par utiliser B qui se trouve dans l'accumulateur droit pour former l'ordre 19 B Z qui sera transformé plus loin. Ensuite, le nouveau mot 1 de la 1ère carte à perforer est formé : 00 n 0003. Le n qui se trouve en 0002 est utilisé pour former l'ordre 19 E n + 2 ainsi que l'ordre 69 n + 2 GA destiné à aller chercher l'ordre de décalage pour l'envoyer en P 0004 = 1980 (afin qu'il soit, pour l'exécution en langage machine, placé en n + 2). En REUN1 = 1596 (n° 161), ce programme rejoint la branche relative à l'ordre de multiplication sans décalage, 03.

- La mémoire EDOGA = 1610 (n° 423) contient 69 0000 1500.

.../...

θ) P' = 4 : division sans décalage (avec index)

178	F 0004	A AD	D ROW2	0 0	00 B 0000	64 B W2	64 0000 W2	
179		T RD	W 0003				64 B W2	en W3 (puis 1+2)
180		R AG	C 0001	0 0A	0 0			
181		D AG	0004	00 A 0000				
182		A AG	O OW1	R EUN2	00 A W1	0 ... 0 W1		
183	R EUN2	A AG	R D00	M UD11	65 A W1	650 0		
164	(M UD11)	T RG	P 0005			65 A W1		en P5 (puis 1+3)
	:	:	:	:	:	:	:	:

- En REUN2 = 1791 (n° 183) les programmes relatifs aux divisions 04 et 06 se rejoignent.
- En MUD11 = 1691 (n° 164), ce programme rejoint celui relatif aux 2 ordres de multiplication, pour la formation de 20 C N et son envoi en W 0002.
- La mémoire DROW2 = 1453 (n° 425) contient 64 0000 W0002 = 64 0000 1992
- La mémoire RD00 = 1794 (n° 429) contient 65 0000 0000.

λ) P' = 6 : division avec décalage (avec index)

184	F 0006	A AD	D ROW2	0 0	00 B 0000	64 B W2	64 0000 W2	
185		T RD	W 0003				64 B W2	en W3 (puis 1 + 2)
186		R AD	P 0001	0 0	00 n 0002			
187		A AD	O 01		00 n 0003	0 01		
188		T RD	P 0001			00 n 0003		en P1 (mot 1)
189		A AD	O 20		00 n+2 0003			

.../...

190		E	DI	E	DOGB		0.....0	00 n+2 0003	69 xxxx GB	
191		S	BF	E	DOGB	18001			69 n+2 GB	en EDOGB = 1392
	(8001)	E	DI		n + 2	G B			3x 000m W1	W1 = 1991
192	G B	T	DI	P	0004		...	...	3x 000m W1	en P4 (puis n+2)
193		D	AD		0004			0....0 m+2		
194		D	AG		0006			n+2 000000		
195		A	AG	C	0001		0.....0 A			
196		D	AG		0004	R EUN2	00 A n+2	0.....0		
183	(R EUN2)	A	AG	R	DOO	M UD11	65 A n+2		650.....0	
164	(M UD11)	T	RG	P	0005				65 A n+2	en P5 (puis 1+3)
	:	:	:	:	:	:				
	:	:	:	:	:	:				

- La 1ère instruction formée est 64 B W 0002 = 64 B 1992. Cette instruction pourrait être formée dans une partie commune aux 2 ordres de division, mais il est plus simple d'utiliser dès le début le mot 00 B 0000 qui est dans l'accumulateur. C'est ensuite le nouveau mot 1 de la 1ère carte qui est formé, puis l'ordre 69 n + 2 GB = 69 n + 2 1550 allant chercher l'ordre de décalage 3x 000m 1991. L'adresse n + 2 sert aussi à former le mot 65 A n + 2. En REUN2 = 1791 (n° 183) ce programme rejoint celui relatif à la division sans décalage.
- La mémoire EDOGB = 1392 (n° 423) contient 69 0000 GB = 69 0000 1550. En fait, l'adresse facteur est modifiée mais n'a aucune importance.

.../...

M) P' = 7 : test SI (avec index)

97	F 0007	A AD C 0001	0.....0	00 B 0000		
98		A AD 4 400		00 B A		
99		T RD P 0005		44 D A	440.....0	
00		R AD M EMNN	0.....0	44 B A		en P5 (puis 1+3)
01		A AG C 0003	0.....0 C	0.....0 N		
02		D AG 0004	00 C 0000	00 N 0000		
03		A AG R 000	60 C 0000		60.....0	
04		A AD C TRL		00 N 1		
05		A AD 4 603		46 N 1+3	46 0000 0003	
06		E DI F 7F8			20 W1 VAPF4	
07		20 W 0001 V APF4				
09	F 7F8	T DI W 0005			20 W1 VAPF4	en W5 = 1995
20		T RD W 0003			46 N 1+3	en W3 (puis 1+2)
21		A AG C TRL	60 C 1			
22		A AG O 02	60 C 1+2			
23		T RG W 0002			60 C 1+2	en W2 (puis 1+1)
24		R AD M EMN	0.....0	00 n 0000		
25		D AG 0002		n 00 0000		
26		A AG C 0005	0.....0 k	n 0.....0		
27		A AD C TRL	0.....0 k	n 00 1		
28		E DI D AG4			35 0004 xxxx	
29		S BI P 0003 D AG4			35 0004 1	en P3 (puis n+1)
30	D AG4	D AG 0004	00 k n	00 1 0000		
31		A AG R 0001	60 CTRk n+1		60 C 0000 0001	= 60 1967 0001
32		T RG P 0002			60 CTRk n+1	en P2 (puis n)
33		A AD W 0005		10 1+1 8003		
34		10 0001 8003				
(W 0005)	T RD W 0001 V APF4				10 1+1 8003	en W1 (puis 1)

.../...

- La 1ère partie de ce programme (jusqu'à F7F8) comporte la construction des mots qui sont utiles pour l'ordre 07 mais pas pour l'ordre 08.
- La 2ème partie (à partir de F7F8) est commune aux ordres 07 et 08. Elle envoie à leur place certains des mots construits par la 1ère partie et fabrique les mots utiles à la fois pour les deux tests (SI et SD). Elle est construite en forme de sous-programme : il faut placer une instruction de sortie dans le distributeur avant d'arriver en F7F8 = 1503 (n° 219).
- Il vaut mieux faire exécuter le plus de choses possibles par la partie commune. C'est pour cela que 46 N 1 + 3 est encore dans l'accumulateur lorsqu'on arrive en F7F8 et que le mot 60 C 0000, est complété par cette partie commune.
- L'ordre F7F8 = 1503 (n° 219) envoie l'instruction de sortie en W 0005 = 1995
- k est placé en C 0005 = 1972 par la partie commune aux ordres à 2 mémoires (ordre n° 67).
- Pour former le mot 35 0004 l par un SBI, la machine a besoin d'entrer dans le distributeur (instruction n° 223) n'importe quel mot commençant par 35 0004, par exemple le mot contenu en DAG4 = 1743 (n° 230). L'instruction DAG 4 n'est pas modifiée.
- La mémoire "4400" = 1476 (n° 208) contient 44 0000 0000.
- La mémoire "4603" = 1442 (n° 209) contient 46 0000 0003.
- La mémoire RGCO1 = 1406 (n° 235) contient 60 00000 0001 = 60 1967 0001.
C 0000 = 1967 est l'adresse du registre d'index n° 0 : CTR0.

v) P' = 2 : test SD (avec index)

					0 0	00 B 0000	
210	F 0008	A AD	M EMNN			00 000B N	
211		A AG	C 0003		0 0 C		
212		D AG	0004		00 C 0000	0B N 0000	
213		A AG	6 9		69 C 0000		690 0
214		A AD	C 0001			0B N A	
215		A AD	9 00			9B N A	90 0
216		E DI		F 7F8			20 W1 VAPF3
217		20 W	0001	V APF3			
219	(F 7F8)	T DI	W 0005			20 W1 VAPF3	en W5 = 1995
220	()	T RD	W 0003			9B N A	en W3 (puis 1 + 2)
221	()	A AG	C TRL		69 C 1		
222	()	A AG	O 02		69 C 1+2		
223	()	T RG	W 0002			69 C 1+2	en W2 (puis 1 + 1)

- La mémoire 69 = 1594 (n° 407) contient 69 0000 0000

- La mémoire 900 = 1526 (n° 213) contient 900000 0000

- Ce programme rejoint celui relatif au test SI en F7F8 = 1503

(n° 219). Voir le paragraphe ci-dessus, relatif à l'ordre 07.

- L'ordre de sortie envoyé en W 0005 = 1995 est ici 20 W 0001 VAPF3 et non 20 W0001 VAPF4 comme pour l'ordre 07. En effet la 2ème carte à perforer ne doit porter que q = 3 mots en plus du mot 1.

X - E - 3) P = 1, 2, 3, 4, 5 : sous-programmes à 3 adresses.

Le facteur de translation $\Delta_{pp'}$ de chaque sous-programme à 3 adresses doit figurer dans la mémoire $DEL + PP' = 1875 + PP' = D 00PP' - 0010$ au moment de l'autoprogrammation. Ainsi, pendant l'autoprogrammation, les facteurs de translations doivent être enregistrés mais pas les sous-programmes, tandis qu'en exécution en langage machine, les sous-programmes doivent être enregistrés mais pas leurs facteurs de translation.

Pour les ordres d'entrée dans les sous-programmes à 3 adresses non indexés, l'autoprogrammation fournit le programme suivant (voir la normalisation au paragraphe VII - C - 3) :

n	RAD	1	n + 1	0 0	00 A B	
n + 1	EDI	1 + 1	$\Delta_{pp'}$			20 C N
1	00	A	B			
1 + 1	20	C	N			

Pour ces mêmes ordres indexés, comme il est noté au paragraphe X - A - 3, l'autoprogrammation fournit le programme :

n	RAD	n + 1	$\Delta_i + 48$
n + 1	60	1	$\Delta_{pp'}$
1	10	000j	000k
1 + 1	00	A	B
1 + 2	20	C	N

.../...

Comme en logique extérieure, les branches relatives à $P = 1, 2, 3, 4, 5$ se rejoignent aussitôt. Les mémoires $G\ 0001 = 1857$, $G\ 0002 = 1858, \dots, G\ 0005 = 1861$ contiennent le même mot ; ce mot est la 1ère instruction de la partie propre aux sous-programmes à 3 adresses. Cette partie débute par un morceau de programme commun aux ordres indexés et non indexés.

359	G 0001	R AD C 0002	S UITG						
360	G 0002	R AD C 0002	S UITG						
	:								
	:								
	:								
363	G 0005	R AD C 0002	S UITG	0.....0	00 B 0000				
364	S UITG	D AG 0002			B 00 0000				
365		A AG C 0001		0....0 A					
366		D AG 0004		00 A B	0.....0				
367		T RG W 0001				00 A B		en W1 (puis 1	
368		R AD P PPRI		0.....0	0.... OPP'			si i = j = k = 0)	
369		D AG 0004			00 00PP' 0000				
370		A AD E DDEL			69 00PP' HA	69 D0000 HA			
371		S AD O DIXO	8002		69 DEL+PP' HA	00 0010 0000			
	8002	E DI DEL+PP'	H A			0.....0 $\Delta_{pp'}$			
372	H A	T DI W 0004				0.....0 $\Delta_{pp'}$		en W4	
373		R AD C 0006		0.....0	00 C N				
374		A AD T D00			20 C N	20.....0			
375		A AG C 0004		i0.....0					
376		A AG C 0005		i0000j000k		00000j000k			
377		G NN I ND3A	N IN3A					index ?	

378	I ND3A	E DI W 0001	io 000j 000k	20 C N	00 A B	
379		T DI W 0002			00 A B	en W2 (puis 1 + 1)
380		T RG W 0001			io 000j 000k	en W1 (puis 1)
381		T RD W 0003			20 C N	en W3 (puis 1 + 2)
382		R AG C TRL	0.....0 1	0.....0		
383		D AG 0004	00 1 0000			
384		A AG W 0004	00 1 $\Delta_{pp'}$		0.....0 $\Delta_{pp'}$	
385		A AG R G00	60 1 $\Delta_{pp'}$		60.....0	
386		A AD D 0052		0.....0 Δ_i		
387		A AD M EMY		0.....0 Δ_{i+48}	0.....048	
388		A AD M EMN		00 n Δ_{i+48}		
389		A AD R D10		65 n+1 Δ_{i+48}	65 0001 0000	
390		T RD P 0002 T GP33			65 n+1 Δ_{i+48}	en P2 (puis n)
391	T GP33	T RG P 0003 V APF3			60 1 $\Delta_{pp'}$	en P3 (puis n + 1)
392	N IN3A	T RD W 0002	0.....0	20 C N	20 C N	en W2 (puis 1 + 1)
393		R AD M EMN	0.....0	00 n 0000		
394		D AG 0002		n 00 0000		
395		A AG C TRL	0.....0 1		0.....0 1	
396		A AD 8001		n 00 1		
397		D AG 0004	00 1 n	00 1 0000		
398		A AG R D01	65 1 n + 1		65 0000 0001	
399		T RG P 0002			65 1 n + 1	en P2 (puis n)
400		A AD W 0004		00 1 $\Delta_{pp'}$	0.....0 $\Delta_{pp'}$	
401		A AD E D10 T DP32		69 1+1 $\Delta_{pp'}$	69 0001 0000	
358	(T DP32)	T RD P 0003 V APF2			69 1+1 $\Delta_{pp'}$	en P3 (puis n + 1)

- La partie commune forme le mot 00 A B qu'elle envoie en réserve en W 0001 = 1991. Pour les ordres non indexés, ce mot restera en W 0001 où il est bien placé ; pour les ordres indexés, il sera envoyé en W 0002. La partie commune a aussi pour rôle d'aller chercher le facteur de translation du sous-programme. Pour cela, elle forme l'ordre 69 DEL + PP' HA = 69 (D00PP' - 0010) HA, puis l'exécute. La mémoire EDDEL = 1730 (n° 448) contient 69 D0000 H A = 69 1885 1656. Le facteur de translation Δ_{pp} est envoyé en W 0004 = 1994 pour être mis en réserve. Ensuite, la partie commune forme 20 C N puis i0 000j 000k qui est testé pour savoir si l'ordre est indexé. S'il est indexé, le programme est aiguillé en IND3A = 1806 (n° 378). Pour former l'adresse $\Delta_i + 48$, la machine va chercher Δ_i en D 0052 = DEL + 62 = 1875 + 62 = 1937 (voir X - A) et lui ajoute 00 0000 0040 placé dans la mémoire MEMY = 1414 (n° 440). Si l'ordre n'est pas indexé, le programme est aiguillé en NIN3A = 1660 (n° 392). L'ordre TDP 32 = 1805 (n° 358) qui termine cette branche est utilisé aussi pour les ordres indexés d'entrer dans les sous - programmes à 2 adresses.

X - E - 4) P' = 9 : bouclage :

Le rôle de l'autoprogrammation pour un ordre de bouclage est de construire des instructions qui, lors de l'exécution en langage machine du programme, permettront d'entrer dans le sous-programme de bouclage. Après la perforation de ces cartes, l'instruction suivante est prise en N.

.../...

Le sous-programme de bouclage figure à poste fixe dans les paquets verts à joindre aux cartes autoperforées. Son facteur de translation a été fixé à 1751 pour l'exécution en langage machine. Pendant l'autoprogrammation, ce facteur est placé sous forme 00 0000 Δ_{bou} dans la mémoire D 0051 = DEL + 61 = 1875 + 61 = 1936.

a) Instructions produites par l'autoprogrammation :

n	RAG	1	1+1
n + 1	00	0000	H
1	69	1 + 1	Δ_{bou}
1 + 1	EDI	n + 1	Δ_{bou}
1 + 2	00	C	N
1 + 3	0x	CTri	B

Ce morceau de programme était primitivement plus simple. Il était de la forme :

n	RAG	n + 1	8003
n + 1	69	1	Δ_{bou}
1	00	0000	H
1 + 1	00	C	N
1 + 2	0x	CTri	B

Mais cette forme ne se prêtait pas à la modification de bouclage. En effet pour que H puisse être modifié par un ordre de modification de bouclage, il faut que cet ordre puisse "savoir" où est H. Or, en programmant, on ne peut pas connaître la valeur de 1 qui sera attribuée à l'ordre de bouclage à modifier. Il fallait donc que H soit en n ou en n + 1. Mais n doit contenir le 1er ordre à exécuter pour le bouclage. Donc H ne pouvait être mis

.../...

qu'en $n + 1$. C'est ainsi que la 1ère forme de programme a été adoptée.

b) Autoprogrammation de l'ordre de bouclage 9i N α H OB OC

108	G 0009	R AG C TRL	0.....01	0.....0	0.....01	
109		D AG 0004	00 1 0000			
110		A AG 8001	00 1 1			
111		A AG R G01	60 1 1+1		60 0000 0001	
112		T RG P 0002			60 1 1+1	en P2 (puis n)
113		A AD M EMN		00 n 0000		
114		A AD D C051		00 n Δ_{bou}	0....0 Δ_{bou}	
115		A AD E D10		69 n+1 Δ_{bou}	69 0001 0000	
116		T RD W 0002			69 n+1 Δ_{bou}	en W2 (puis 1 + 1)
117		D AD 0006	000000xxxx	xx 1+1 xxxx		
118		S BF W 0001			69 1+1 Δ_{bou}	en W1 (puis 1)
119		E DI C 0006			00 C N	
120		T DI W 0003			00 C N	en W3 (puis 1 + 2)
121		R AG P PPRI	0.....09i	0.....0		
122		D AD 0002	0.....0	9i 0.....0		
123		A AD C 0002		9i B 0000	00 B 0000	
124		D AD 0003		0009i B 0		
125		A AD C 0004		α 009i B 0	α 0.....0	
126		D AD 0001		0 α 009i B		
127		A AD M CTR0		0 α CTRi B	00 1877 0000	
128		T RD P 0005			0 α CTRi B	en P5 (puis 1 + 3)
129		E DI C 0001			0.....0 H	
130		T DI P 0003 V APF4			0.....0 H	en P3 (puis n + 1)

- Comme il est indiqué au paragraphe X - E - 4, la mémoire D 0051 = $DEL + 61 = 1875 + 61 = 1936$ contient 00 0000 $\Delta_{bou} = 00\ 0000\ 1751$,
- La partie commune aux ordres à 2 mémoires a placé 0.....0PP' = 0.....09i en PPPRI = 1360 et $\alpha\ 0.....0$ en C 0004 = 1971.
- Pour former l'adresse CTRi, on a ajouté à PP' = 9i la constante CTRI - 90 = $1967 - 90 = 1877$. La mémoire MCTRO = 1486 (n° 406) contient 00 1877 0000.

Remarque : dans l'assemblage de l'autoprogrammation, avant d'avoir sa forme actuelle (ci-dessus) la partie relative au bouclage était différente. Or on a omis d'enlever l'ancienne forme que représentaient les instructions n° 89 à 107. Ceci n'a aucune importance d'ailleurs car l'encombrement de l'autoprogrammation ne compte pas. Dans la liste figurant à la fin de cette brochure, les cartes inutiles ont été supprimées.

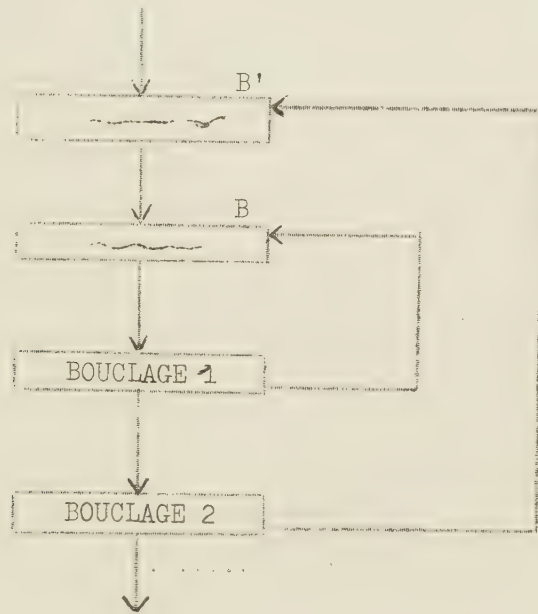
c) Sous-programme de bouclage pour l'exécution en langage machine.

L'organigramme de ce sous-programme de bouclage est semblable à celui du sous-programme utilisé en logique extérieure, à l'exception de deux points :

- 1) pour les ordres de bouclage tels que $\alpha = 0$, après le 1er passage la phase de formation des ordres est supprimée. Ceci donne un peu plus d'encombrement au sous-programme mais provoque un gros gain de temps. La suppression de la phase de formation des ordres se fait en remplaçant le 1er ordre du sous-programme de bouclage (placé en $\Delta_{bou} + 0000 = 1751$) par l'ordre "aller à BØU2 = $\Delta_{bou} + 35 = 1751 + 35 = 1786$ " (voir l'organigramme du paragraphe VII - C - 4).

Seuls peuvent être dotés de $\alpha = 0$ les bouclages ne comportant pas de bouclage intérieur.

Contre-exemple : considérons 2 bouclages intérieurs l'un à l'autre dont le plus extérieur soit affecté de $\alpha = 0$.



Bouclage 1 : $9i \ N \ 1 \ H \ OB \ OC$

Bouclage 2 : $9i' \ N' \ OH' \ OB' \ OC'$

La 1ère fois que l'on passe par le sous-programme de bouclage, c'est pour effectuer le bouclage 1. Celui-ci se déroule normalement, sans suppression de la formation des ordres. Quand les H tours du bouclage 1 ont été effectués, le sous-programme de bouclage est utilisé pour le 1er passage par l'ordre de bouclage 2. Après ce 1er passage, la phase de formation des ordres est détruite. Lorsqu'on repasse au bouclage 1, le sous-programme de bouclage (ne formant plus les ordres nécessaires) utilise les ordres formés précédemment, c'est-à-dire ceux construits

.../...

pour le bouclage 2. Cela n'est pas correct et le bouclage 1 ne fonctionne pas comme il devrait le faire.

2) pour les bouclages tels que $\alpha = 0$, à la sortie de boucle (après la remise à zéro du compteur de boucles n° 0 et du registre d'index n° i) : la mémoire $\Delta_{\text{bou}} + 0000 = 1751$ est régénérée ; la phase de formation des ordres est donc rétablie.

Remarque : l'ordre - 02 U V fait aussi cette régénération.

Voici le sous-programme de bouclage destiné à l'exécution en langage machine.

Ce programme a été écrit en langage machine. Chaque ordre est suivi de ses indicatifs de translation.

n	60	1	1 + 1		69 1+1 Δ	0	0		$\Delta = \Delta_{\text{bou}}$
1 + 1	69	n + 1	Δ				0	0 H	
$\Delta +$	0000	24	0055	0001	8 8		0	0 H	en $\Delta + 55$
	1	11	0002	8003	8 9	15 1+2 $\Delta + 5$			
	2	53	9998	9995	9 9				
	(8003)	15	1+2	0005	8		00	C N	
	5	69	0042	0006	8 8		24	Δ xxxx	
	6	23	0042	0003	8 8		24	Δ N	en $\Delta + 42$
	3	20	0004	0007	8 8		00	C N	en $\Delta + 4$
	7	10	0008	8003	8 9	65 1+3 $\Delta + 9$			
	8	50	0001	0004	9 9				
	(8003)	65	1+3	0009	8	0	0	Δ CTRi B	
	9	69	8003	0010	9 8		0	0	
	10	22	1991	0011	9 8		00	CTRi 0000	en W 0001

.../...

0011	10	8001	0012	9 8	00 CTRi 0000	0α CTRi B	00 CTRi 0000	
12	11	0013	0014	8 8	00 i 0000			
13	00	1967	0000	9 9				
14	44	0017	0015	8 8				i ≠ 0 ?
15	15	0016	0017	8 8		0α 1991 B		Cas où i = 0
16	00	0024	0000	9 9				
17	69	0047	0018	8 8		0α CTRi 1991 B	10 xxxx Δ+ 49	
18	22	0047	0019	8 8			10 CTRi 1991 Δ+ 49	en Δ + 47
19	69	0049	0020	8 8			21 xxxx xxxx	
20	22	0049	0021	8 8			21 CTRi 1991 xxxx	
21	23	0049	0022	8 8			21 CTRi 1991 B	en Δ + 49
22	69	0038	0023	8 8			20 xxxx Δ+ 39	
23	22	0038	0024	8 8			20 CTRi 1991 Δ+ 39	en Δ + 38
24	11	8003	0025	9 8	0		00 i 0000	
25	30	0008	0026	9 8		00 α		
26	35	0004	0050	9 8		00 000α 0000		
50	45	0051	0052	8 8				α ≠ 0 ?
51	69	0041	0054	8 8			24 Δ+55 Δ+1	cas où α ≠ 0
52	69	0053	0054	8 8			00 0000 Δ+35	cas où α = 0
53	00	0000	0035	9 8				
54	24	0000	0027	8 8			24 Δ+55 Δ+1	
							00 0000 Δ+35	en Δ + 0
27	15	0028	0029	8 8		00 CTRHα 0000		
28	00	1940	0000	9 9				
29	69	0035	0030	8 8			60 xxxx Δ+44	
30	22	0035	0031	8 8			60 CTRHα Δ+ 44	en Δ + 35
31	69	0039	0032	8 8			24 xxxx Δ+40	

.../...

0032	22	0039	0033	8	8		00 CTRH α 0000	24 CTRH Δ + 40	en Δ + 39
33	69	0046	0034	8	8			21 xxxx Δ + 36	
34	22	0046	0035	8	8			21 CTRH α Δ + 36	en Δ + 46
(35)	60	CTRH α	0044	8		0 0 h	0 0		
44	10	0045	0046	8	8	0 ... 0 h+1			
45	00	0000	0001	9	9				
(46)	21	CTRH α	0036	8				0 0 h+1	en CTRH α
36	11	0055	0037	8	8	0 ... 0 h+1-H		0 0 H	
37	46	0048	0038	8	8				h + 1 - H < 0 ?
{ (38)	20	CTRi	0039	8				0 0	en CTRi (si i $\neq$ 0)
{ (38)	20	1991	0039	8				0 0	en 1991 (si i = 0)
39	24	CTRH α	0040	8				0 0	en CTRH α
40	69	0041	0042	8	8			24 Δ + 55 Δ + 1	
41	24	0055	0001	8	8				
(42)	24	0000	N	8				24 Δ + 55 Δ + 1	en Δ + 0
48	60	0004	0043	8	8	00 C N	0 0		
43	30	0004	0047	9	8	0 ... 0 C	N 00 0000		cas où h+1 < H
{ (47)	10	CTRi	0049	8		0 ... 0 (CTRi) + C		0 ... 0 (CTRi)	si i $\neq$ 0
{ (47)	10	1991	0049	8		x x		x x	si i = 0
{ (49)	21	CTRi	B			0 ... 0 (CTRi) + C		0 ... 0 (CTRi) + C	en CTRi (si i $\neq$ 0)
{ (49)	21	1991	B			x x		x x	en 1991 (si i = 0)

.../...

- Les instructions $\Delta + 0$ à $\Delta + 34$ constituent la partie de formation des ordres ; cette formation a lieu à partir des indications données par les mots qui ont été fournis par l'autoprogrammation. D'abord 00 0000 H est envoyé dans la mémoire $\Delta + 55$. Ce n'est pas dans une mémoire de travail qu'il est envoyé car il doit être conservé d'un passage à l'autre si la phase de formation des ordres est supprimée. Il en est de même de 00 C N envoyé en $\Delta + 4$ et non dans une mémoire de travail.
- Pour tester i, on le forme dans l'accumulateur gauche. Pour cela, on appelle 00 CTRi 0000 dans cet accumulateur et on en soustrait la quantité 00 1967 0000 (1967 est l'adresse du CTR0). Si la réponse au test est non, $i = 0$, l'accumulateur droit contient 0α CTR0 B = 0α 1967 B. Pour éviter de former des ordres susceptibles de modifier le CTR0, on remplace l'adresse facteur de l'accumulateur droit par W 0001 = 1991 en lui ajoutant 00 0024 0000 : $1967 + 24 = 1991$. Ainsi les ordres formés à partir de cette adresse ne modifieront pas la mémoire CTR0 mais la mémoire W 0001 qui peut être transformée sans inconvénient.
- L'ordre d'adresse $\Delta + 50$ exécute un test sur α . Si $\alpha = 0$, l'ordre $\Delta + 52$ prépare dans le distributeur le mot 00 0000 $\Delta + 35$ que l'ordre $\Delta + 54$ envoie en $\Delta + 0$ pour que la phase de formation des ordres soit supprimée (pour le prochain passage par l'ordre de bouclage). Si $\alpha \neq 0$, l'ordre $\Delta + 54$ l'envoie en $\Delta + 0$ (ce qui est d'ailleurs inutile).

.../...

- On forme l'adresse du compteur de boucles $n^\circ \alpha$, CTRH α , en ajoutant à 00 0000 α 0000 la constante 00 CTRH0 0000 = 00 1940 0000.
 - En $\Delta + 35$ débute l'exécution proprement dite du bouclage.
 - L'instruction $\Delta + 37$ exécute un test sur $h + 1 - H$. Si $h + 1 - H \geq 0$ le programme est aiguillé en $\Delta + 38$ pour exécuter la sortie de boucle.
- Le registre d'index CTRi et le compteur de boucles CTRH α sont remis à zéro. En fait, si $i = 0$ c'est la mémoire 1991 = W 0001 qui est remise à zéro. Ensuite a lieu la régénération de la mémoire $\Delta + 0000$, indispensable dans le cas où $\alpha = 0$. Le dernier ordre adresse à l'instruction N.
- Si $h + 1 - H < 0$, le programme est aiguillé en $\Delta + 48$ pour que le bouclage continue. La machine ajoute C au contenu du registre d'index CTRi si $i \neq 0$ puis adresse à B pour boucler.

X - F - SOUS-PROGRAMMES PLACES A POSTE FIXE.

Comme en logique extérieure, deux sous-programmes ont été placés à poste fixe dans les paquets verts à joindre aux programmes autoperforés. Ce sont la perforation d'une séquence de mémoires et la modification de bouclage.

X - F - 1 - Perforation (ordre 49 N iA jB kC)

L'entrée dans ce sous-programme est standard, et est traitée en autoprogrammation comme l'entrée dans n'importe quel sous-programme. Dans les paquets d'autoprogrammation a été placée à poste fixe une carte chargeant

00 0000 $\Delta_{49} = 00 0000 1889$ en $DEL + 49 = 1875 + 49 = 1875 + 49 = 1924$.

L'organigramme est semblable à celui qui a été fait en logique extérieure, à part que la branche FIN envoie, après la perforation, à l'adresse N et non à DEBUT = 1800.

Le programme est légèrement plus long que celui en logique extérieure car ce dernier, non standard, avait une partie de formation d'ordres moins importante. Il a été écrit en PASCAL puis assemblé, perforé sous forme translatable et traduit en $\Delta_{49} = 1889$.

1	0000	T DI	W 0001		0 0	00 A B	20 C N	Entrée standard
2		E DI	6 9AG				20 C N	en W1 = 1991
3		D AG	0004		0 0	AA A A B 0000	69 0000 8003	
4		S BF	W 0002				69 B 8003	en W2 = 1992
5		D AD	0008		0 0	0 0A		
6		T RD	W 0003				0 0 A	en W3 = 1993
7		D AG	0004			00 A 0000		
8		A AD	2 4PS1			24 P2+A SUIT1	24 P2 SUIT1	
9		T RD	W 0004				24 P2+A SUIT1	en W4 = 1994
10		R AD	W 0001		0 0	20 C N		
11		E DI	P FFI				71 P1 xxxx	
12		S BI	P FFI				71 P1 N	en PF FFI
13		E DI	8003				0 0	
14		S BF	W 0005				00 C 0000	en W5 = 1995
15		R AD	W 0002	B UCL	0 0	69 B 8003		
16	B UCL	A AG	2 4PS1		24 P2 SUIT1	69 X 8003		
17		E DI	W 0003				0 0 A	
18		S BF	P 0001	8002			00 X A	en P1 = 1977
	(8002)	E DI	X	8003			(X)	
	(8003)	T DI	P 0002	SUIT1			(X)	en P2 = 1978

.... /

19	S,UIT1	A AG	O 10		24P2+m+1 SUI1	69 X+m 8003		
20		S AD	W 0005			$\begin{smallmatrix} 8 \\ 9 \end{smallmatrix}$ X+m-C 8003	00 C 0000	
21		E DI	8002				$\begin{smallmatrix} 8 \\ 9 \end{smallmatrix}$ X+m-C 8003	
22		S D9		F IN				X+m < C ?
23		A AD	O 10			68 X+m-C+1 8003		
24		A AD	W 0005			69 X+m+1 8003		
25		S AG	W 0004		m+1-A		24 P2+A SUI1	
26		G NN		P FØ				m+1 ≠ A ?
27		A AG	8001	8002	24P2+m+1SUI1			
	(8002)	E DI	X +m+1	8003			(X+m+1)	
	(8003)	T DI	P2+m+1	S UI1			(X+m+1)	en P2 + m + 1
28	P FØ	P FØ	P 0001	B ØUCL	O.....0	69 X+m+1 8003		
29	F IN	R AD	8003		O.....0	24 P2+m+1 SUI1		
30		S AD	2 4PS1			00 m+1 0000		
31		E DI	P 0001				00 X A	
32		D AD	0004			O.....0 m+1		
33		S BI	P 0001	P FØFI			00 X m+1	en P1 = 1977
	(P FØFI)	P FØ	P 0001	I N				
34	O 10	00	0001	0000				
35	P FØFI	71	P 0001	0000				
36	2 4PS1	24	P 0002	S UI1				
37	6 9AG	69	10000	8003				

.../...

L'ordre n°12, après avoir formé l'ordre final PFØ P0001 N, l'envoie à sa place, c'est - à - dire en PFØFI. On aurait d'ailleurs pu gagner la mémoire PFØFI en remplaçant les ordres n° 11 et 12 respectivement par EDI PFØ et SBI W0006, et en remplaçant l'adresse "instruction suivante" de l'ordre n°33 par W0006.

X - F - 2 - Modification de bouclage (ordre 00 N 01937 jB kC)

La modification de bouclage doit remplacer le H de l'ordre de bouclage placé en C et C+1 par H', placé sous la forme 00 0000 H' dans la mémoire B. En exécution en langage machine, la valeur de H correspondant à l'ordre de bouclage d'adresse C est placée sous la forme 00 0000 H dans la mémoire C+1 (voir le paragraphe X-E-4-a). Pour effectuer la modification de bouclage, il suffit d'envoyer 00 0000 H' en C+1.

Le facteur de translation de ce court sous - programme a été fixé à 1885. Mais, pour les raisons expliquées au paragraphe VII - D - 2, les deux premières mémoires ont été remplacées par 1937 et 1938 et ainsi libérées. Le sous - programme de modification de bouclage destiné à l'exécution en langage machine peut donc s'écrire :

				65 B A+1	20 C N		
A =	1937	15	Δ+2	Δ+3	24 C+1 N	04 0001 0000	
Δ +	0002	04	0001	0000			
	0003	20	1938	8003		24 C+1 N	en A + 1 = 1938
	(8003)	65	B	A + 1	0....0	0....0 H'	00 0000 H'
A+1=	(1938)	24	C+1	N		00 0000 H'	en C + 1

XI - DISPOSITION DE L'AUTOPROGRAMMATION

ET DES SOUS-PROGRAMMES DES " PAQUETS VERTS "

XI - A - AUTOGRAMMATION :

- Le programme d'autoprogrammation occupe les mémoires 1352 à 1881.
- Les mémoires 1882, 1883, 1884 sont respectivement les mémoires CTRL, LMAX et MEMNN (qui contient l'adresse de l'instruction suivante). La mémoire 1800 contient la 1^{ère} instruction de l'autoprogrammation et est appelée DEBUT.
- Les mémoires $DEL + 10 = 1875 + 10 = D0000$ à $DEL + 59 = 1934 = D0049$ sont destinées à recevoir les facteurs de translation Δ_{PP} .
- $DEL + 60 = 1935$ peut aussi contenir un Δ .
- $DEL + 61 = D0051 = 1936$ contient le facteur de translation du sous - programme de bouclage, soit 00 0000 1751.
- $DEL + 62 = D0052 = 1937$ contient le facteur de translation du sous - programme d'indexage, soit 00 0000 1807.
- Les mémoires 1940 à 1999, utilisées pour la plupart par le programme de chargement, ne peuvent pas contenir d'instruction.
- Certaines des mémoires utilisées comme registres d'index pour l'exécution des programmes (1967 à 1976) jouent en autoprogrammation le rôle de mémoires de travail.
- Les mémoires de perforation 1977 à 1984 jouent un rôle très important. C'est à partir de ces mémoires que se fait la perforation des programmes en langage machine formés par l'autoprogrammation.
- Les mémoires 1991 à 1999 sont utilisées comme mémoires de travail. En particulier, les mots devant aller en mots 2, 3 et 4 des secondes cartes de chaque ordre en CDP sont envoyés d'abord en 1991, 1992, 1993.

XI - B - EXECUTION EN LANGAGE MACHINE (paquets verts)

La partie fixe des paquets (verts) à joindre aux programmes autoperforés est composée de la façon suivante :

1751 1806	Sous - programme de bouclage
1807 1881	Sous - programme triple d'indexage
1882 à 1886	Libres
1887 1888	Sous-programme de modification de bouclage (début en 1937)
1889 1925	Sous - programme de perforation (ordre 49)
1926 à 1929	Libres
K = 1930 1935	Partie constante de l'initialisation (-02 U V)
1936	Mémoire ARETM . Contient : 01 1335 8000
1937 1938	Début de la modification de bouclage
1939	Mémoire Z . Contient : 44 ARETM W0002 = 44 1936 1992
1940 1949	Compteurs de boucles (région H)
1950	Libre
1951 1960	Mémoires de lecture (région L). Utilisables par le programmeur
1961 à 1966	Libres
1967 1976	Registres d'index (région C). 1967 contient toujours ZERO
1977 1986	Mémoires de perforation (région P). Non utilisables par programmeur
1987 1990	Libres. Mais 1987 est utilisé comme mémoire de travail par le sous - programme auxiliaire de décalage.
1991 1999	Mémoires de travail (région W)

XII - SOUS-PROGRAMMES ADAPTÉS AU C.D.P.

DIVERS C.D.P. FORMÉS

XII - A - GENERALITES

A la partie commune du C.D.P. peuvent être ajoutés des sous-programmes à 2 ou 3 adresses. C'est ainsi qu'ont été constitués les trois C.D.P. existant pour le moment : virgule fixe, virgule flottante simple et double précision. Chacun de ces C.D.P. peut être complété par d'autres sous-programmes. D'autres C.D.P. peuvent être créés en ajoutant des sous-programmes à la partie commune du C.D.P.

Les sous-programmes à ajouter au C.D.P. doivent être compatibles avec l'une des 2 formes d'entrée prévues : entrée dans les sous-programmes à 2 ou ceux à 3 adresses. Ils peuvent utiliser les mémoires W 0001 = 1991 à W 0007 = 1997 comme mémoires de travail, c'est-à-dire y envoyer des résultats ou des indications intermédiaires comme, par exemple, l'instruction de sortie.

Rappelons l'état des accumulateurs et du distributeur au moment de l'entrée dans un sous-programme :

- Sous-programme à 2 adresses :

8003			8002			8001		
65	B	A + 1	20	C	N'			

La 1ère instruction est en A.

- Sous-Programme à 3 adresses :

8003	8002	8001
0.....0	00 A B	20 C N'

La 1ère instruction est en $\Delta_{PP'}$.

Les 2 formes d'entrée dans les sous-programmes sont indexables par le C. D. P. lui-même, ce qui leur donne un grand intérêt et une grande souplesse d'utilisation.

Pour adjoindre au C. D. P. un sous-programme à 2 adresses compatible avec lui, il suffit de choisir pour ce sous-programme un facteur de translation valable à la fois en logique extérieure et en exécution en langage machine. Le sous-programme, écrit sous forme translatable 5 mots par carte et précédé de son facteur de translation, doit être joint au paquet de logique extérieure ainsi qu'à celui destiné à l'exécution en langage machine. La 1ère instruction de ce sous-programme, après translation, se trouve alors en A. L'ordre qui commande son exécution est : 00 N 0A jB kC.

S'il est impossible de trouver une place pour le sous-programme, libre à la fois en logique extérieure et en exécution en langage machine, il faut couper le sous-programme en 2 parties :

- 1°) Une partie constituée de 2 mémoires consécutives au moins, dont l'emplacement soit le même en logique extérieure et en langage machine ; c'est dans cette partie que seront choisies les adresses A et A + 1, A devant contenir la 1ère instruction du sous-programme.
- 2°) L'autre partie translatable différemment en logique extérieure et en langage machine. L'exécution du sous-programme sera commandée par l'ordre 00 N 0A jB kC.

Pour adjoindre au C.D.P. un sous-programme à 3 adresses compatible avec lui, il faut choisir pour ce sous-programme non seulement un facteur de translation mais aussi un code PP'.

En logique extérieure, il faut joindre au paquet du C.D.P. le sous-programme translatable précédé de son facteur de translation $\Delta_{pp'}$, et une carte destinée à charger en $DEL + PP' = 1875 + PP'$ le mot $00\ 0000\ \Delta_{pp'}$. En exécution en langage machine, il faut joindre au paquet du C.D.P. et aux cartes autoperforées le sous-programme translatable précédé de son facteur de translation qui peut être différent de celui en logique extérieure (soit $\Delta'_{pp'}$).

Il ne faut pas mettre de carte chargeant $00\ 0000\ \Delta'_{pp'}$ en $1875 + PP'$. Par contre il est indispensable de joindre cette carte au paquet d'autoprogrammation.

Il est inutile de joindre à celui-ci le sous-programme lui-même. L'ordre qui commande l'exécution du sous-programme est : PP' N iA jB kC.

Remarque :

Pour créer des sous-programmes logiques (tests, aiguillages, ...) il faudra prendre de grandes précautions car en général ces sous-programmes ne pourront pas être les mêmes en logique extérieure et en exécution en langage machine. Cela est dû au fait que dans ces 2 procédés la façon de désigner l'adresse de l'instruction suivante N est différente : N doit être placé en MEMNN = 1884, en logique extérieure, et en position d'adresse instruction suivante de la dernière instruction, en langage machine.

XII - B - C. D. P. EN VIRGULE FLOTTANTE SIMPLE PRECISION :

Le C. D. P. en virgule flottante simple précision est formé du tronc commun du C. D. P. auxquels sont joints 3 sous-programmes à 3 adresses et 6 sous-programmes à 2 adresses.

- Les sous-programmes à 3 adresses sont les 4 opérations en virgule flottante FLAIR :

11	N	iA	jB	kC	(A) + (B) en C	} 2 entrées différentes dans le même sous-programme.
12	N	iA	jB	kC	(A) - (B) en C	
13	N	iA	jB	kC	(A) x (B) en C	
14	N	iA	jB	kC	(A) : (B) en C	

Ces sous-programmes ont le même emplacement en logique extérieure et en langage machine :

l'ordre	11	} occupe les mémoires		1487 à 1555
	12			
	13	"	"	1450 à 1486
	14	"	"	1414 à 1449

En logique extérieure et en autoprogrammation, il existe une carte qui charge :

en DEL + 11 = 1886	00	0000	1487
en DEL + 12 = 1887	00	0000	1488
en DEL + 13 = 1888	00	0000	1450
en DEL + 14 = 1889	00	0000	1414

- Les sous-programmes à 2 adresses calculent les fonctions principales.

00	N	0	1000	jB	kC	$\sqrt{(B)}$ en C	} même sous-programme
			1002			sin (B) en C	
			1004			cos (B) en C	
			1006			Arc sin (B) en C	

... ..

... / ...

00 N 0 T008 jB kC	$\log_{10} (B) \text{ en } C$	} même sous-programme
1014	$\text{Log}_e (B) \text{ en } C$	
1010	$10^{(B)} \text{ en } C$	} même sous-programme
1016	$e^{(B)} \text{ en } C$	
1012	$ (B) \text{ en } C$	

Ces sous-programmes occupent respectivement les mémoires suivantes :

$\sqrt{(B)}$	1381 à 1413
sin et cos	1201 à 1288
Arc sin	1101 à 1200
Logarithmes	1020 à 1100
Exponentielles	1289 à 1380
Valeur absolue	1012 et 1013

En fait, pour simplifier les codes de ces ordres, on a reproduit :

1381 et 1382	en	1000 et 1001
1201 et 1202	en	1002 et 1003
1203 et 1204	en	1004 et 1005
1101 et 1102	en	1006 et 1007
1022 et 1023	en	1008 et 1009
1020 et 1021	en	1014 et 1015
1291 et 1292	en	1010 et 1011
1289 et 1290	en	1016 et 1017

Les mémoires 1381, 1382, 1201 à 1204, 1101, 1102, 1020 à 1023 et 1289 à 1292 pourraient être libérées.

.../...

XII - C - C. D. P. EN VIRGULE FLOTTANTE DOUBLE PRECISION. -

Ce C. D. P. est formé du C. D. P. en virgule flottante simple précision auquel sont jointes les 4 opérations en virgule flottante double précision (2, 18). Chaque nombre est contenu dans 2 mémoires ; par exemple le nombre désigné par son adresse D est en fait contenu en D et D + 1. Les 4 opérations sont exécutées par 3 sous-programmes à 3 adresses :

21 N iA jB kC	addition	} même sous-programme
22 N iA jB kC	soustraction	
23 N iA jB kC	multiplication	
24 N iA jB kC	division	

Ces sous-programmes ont le même emplacement en logique extérieure et en langage machine :

{ 21 occupe les mémoires	714 à 840
{ 22	
23 " " "	841 à 914
24 " " "	915 à 999

En logique extérieure et en autoprogrammation, il existe une carte qui charge :

en DEL + 21 = 1896	00 0000 0714
en DEL + 22 = 1897	00 0000 0715
en DEL + 23 = 1898	00 0000 0841
en DEL + 24 = 1899	00 0000 0915

XII - D - C. D. P. EN VIRGULE FIXE. -

Les opérations en virgule fixe existent dans la partie commune du C. D. P.. Seules les fonctions ont été ajoutées à ce tronc commun pour former le C. D. P. en virgule fixe. Mais les sous-programmes de calcul de fonction en virgule fixe exigent une certaine position de virgule pour l'argument et fournissent la fonction

avec une position de virgule bien déterminée. Or, ces positionnements ne sont pas toujours compatibles avec le reste du programme. Il a donc été jugé nécessaire de prévoir des possibilités de décalage : avant l'exécution du sous-programme, décalage de l'argument pour ajuster la position de sa virgule sur la position exigée par le sous-programme ; après l'exécution du sous-programme, décalage de la fonction pour l'amener au positionnement voulu pour le reste du problème.

Ainsi les calculs des fonctions principales en virgule fixe se font par des sous-programmes non pas à 2 adresses mais à 3. Ces 3 adresses sont : l'adresse de l'argument, celle où doit être envoyée la fonction et enfin 4 chiffres désignant les décalages à effectuer sur l'argument et la fonction. Les ordres d'opérations en virgule fixe avec décalages sont de la forme : $5P' N iA jB \alpha \lambda \beta \mu$. P' désigne la nature de la fonction, A l'adresse de l'argument, B celle de la fonction. α et λ définissent le décalage à faire sur l'argument. Ce décalage est, en langage machine : $3\alpha \ 000 \lambda \ xxxx$. De même β et μ définissent le décalage $3\beta \ 000 \mu \ xxxx$ à exécuter sur la fonction avant de l'envoyer en B . Ainsi α et β définissent la nature de chacun des décalages, et λ et μ leur grandeur. Les calculs se font par l'intermédiaire d'un sous-programme nommé "sous-programme auxiliaire de décalage".

XII - D - 1 - Sous-programme auxiliaire de décalage :

C'est un sous-programme à 3 adresses à 10 entrées, une pour chaque valeur de P' . Appelons D le facteur de translation de ce sous-programme.

L'entrée relative à $P' = 0$ se fait en $D + 0$, celle relative à $P' = 1$ se fait en $D + 1, \dots$, celle relative à $P' = 9$ en $D + 9$. Ainsi dans la mémoire $DEL + 50 = 1925$ se trouve $00 \ 0000 \ D$, en $DEL + 51 = 1926$ se trouve $00 \ 0000 \ D+1, \dots$,

.../...

en $DEL + 59 = 1934$ se trouve 00 0000 D+9. Les entrées relatives aux divers P' se rejoignent après que chaque branche ait introduit dans le distributeur le facteur de translation $\Delta_{P'}$ du sous-programme devant calculer la fonction désignée par P' .

Voici la succession des ordres qui précèdent le rassemblement des branches en $D + 30$:

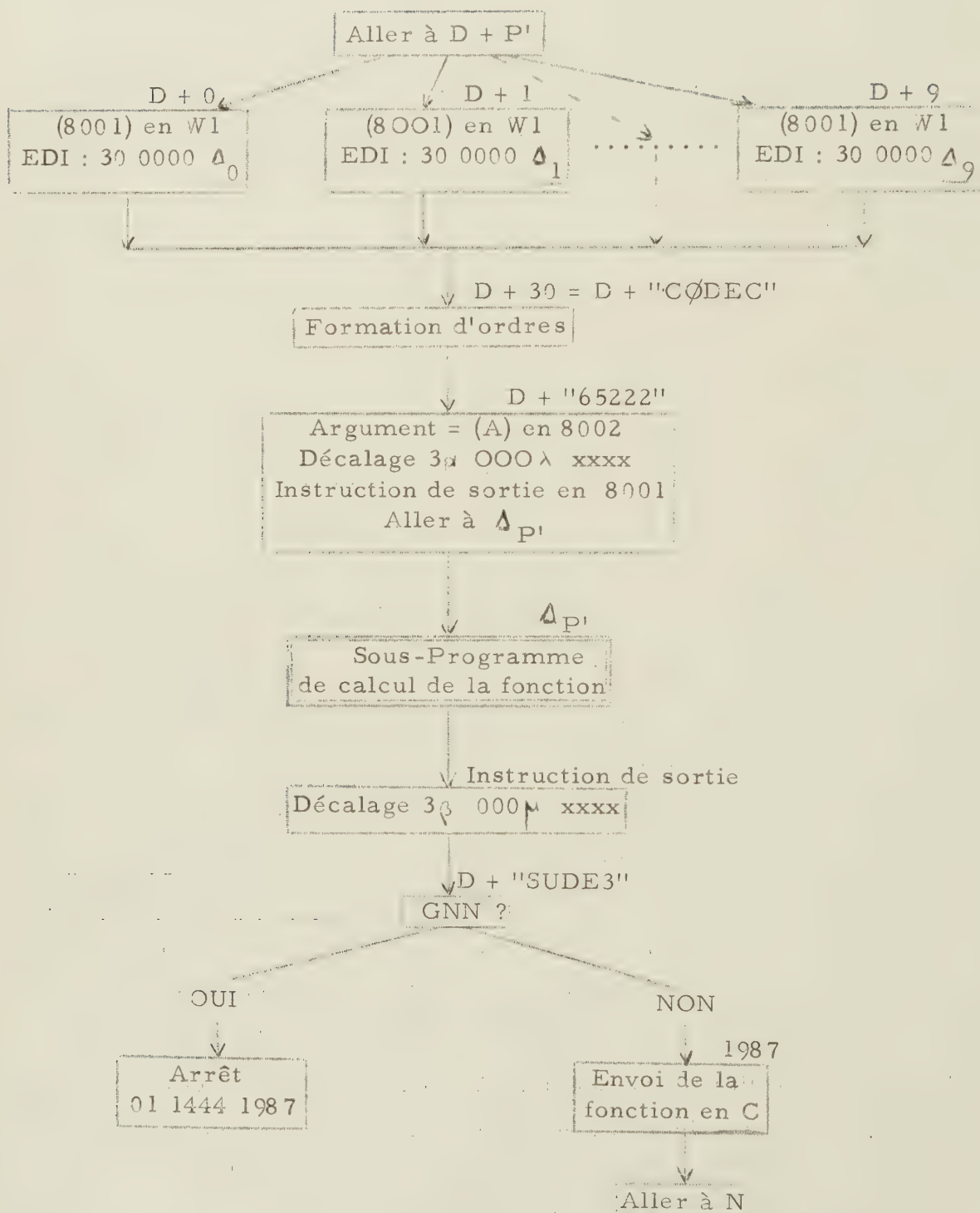
Adresse	Code	F	I	(8001)	
$D + P'$	TDI	W 0001	$D + 10 + P'$	20 $\alpha \lambda \beta M$ N	en W1
$D + 10 + P'$	EDI	$D + 20 + P'$	<u>$D + 30$</u>	20 $\alpha \lambda \beta M$ N	
$D + 20 + P'$	30	0000	$\Delta_{P'}$	30 0000 $\Delta_{P'}$	

Cette partie, existant pour les 10 valeurs de P' , occupe les 30 premières mémoires du sous-programme.

Le déroulement du programme peut se représenter par le schéma suivant :
(voir schéma page suivante)

Les sous-programmes de calcul de fonctions en virgule fixe adaptables au sous-programme auxiliaire de décalage doivent prendre l'argument dans l'accumulateur (gauche, droit ou les deux) et l'instruction de sortie dans le distributeur. Ils remettent la fonction calculée dans l'accumulateur (gauche, droit ou les deux). L'entrée effectuée par le programme auxiliaire de décalage prépare dans l'accumulateur l'argument x décalé et dans le distributeur l'instruction de sortie qui n'est autre que l'ordre de décalage à effectuer sur la fonction avant de l'envoyer en B. Après ce décalage, et avant le transfert de l'accumulateur droit en B, a lieu un test sur l'accumulateur gauche qui provoque l'arrêt 01 1444 1987 si cet accumulateur est non nul, c'est-à-dire si la fonction a plus de 10 chiffres.

.../...



.../...

Les sous-programmes auxiliaires de décalage utilisés en langage machine et en logique extérieure sont un peu différents car dans ce dernier cas le sous-programme comporte la préparation de l'analyse. En effet, une analyse spéciale a été créée pour les ordres de calculs de fonctions avec décalages. Cette analyse permet de mettre en évidence les décalages. Les cartes obtenues sont de la forme :

Mot 1 : n

Mot 2 : }
Mot 3 : } Instruction 5P' N iA jB 0 $\alpha \lambda \beta \mu$

Mot 4 : 0 (CTRi) (CTRj) 000

Mot 5 : }
Mot 6 : } Argument dans l'accumulateur entier après décalage

Mot 7 : Fonction après décalage (telle qu'elle est envoyée en B)

Mot 8 : 20 $\alpha \lambda \beta \mu$ 1700

Les mots 1,2,3,4 sont préparés par la partie commune de la logique extérieure.

a) Sous - programme auxiliaire de décalage pour l'exécution en langage machine :

	P'RA	0000	0029						
	P'RA	0055	1990						
	P'RG	W 1991	1999						
	P'EN	C ØDEC	0030						
				0..... 0	00	A	B	20 $\alpha \lambda \beta \mu$ N	
(D + P')	24	1991	D +10+P'					20 $\alpha \lambda \beta \mu$ N	en W1
(D + 10 + P')	69	D +20+P'	D + 30					30 0000 $\Delta_{P'}$	
(D + 20 + P')	130	0000	$\Delta_{P'}$						

.../...

D + C	DEC	T DI	W 0002	0 0	00 A B	30 0000 Δ_P	en W2
		E DI	6 5222			65 xxxx	
		S BF	6 5222			65 A	en 65222
		D AG	0004	0 0	xx B 0000		
		E DI	W 0001			20 $\alpha \lambda \beta \mu$ N	
		S BF	1987			20 B N	en 1987
		R AD	W 0001	0 0	20 $\alpha \lambda \beta \mu$ N		
		D AD	0004		000020 $\alpha \lambda \beta \mu$		
		D AG	0005	0 02	0 $\alpha \lambda \beta \mu$ 000000		
		A AD	W 0002		3 $\alpha \lambda \beta \mu$ 0 Δ_P		
		R AD	8002	0 0	3 $\alpha \lambda \beta \mu$ 0 Δ_P	3 $\alpha \lambda \beta \mu$ 0 Δ_P	
		D AD	0003		0003 $\alpha \lambda \beta \mu$ 0x		
		S BF	W 0001			3x 0xx $\lambda \Delta_P$	en W1
		D AG	0006	x x	0x000000		
		S AG	8003	0 0			
		D AD	0001		0 $\beta \mu$ 0x000000		
		A AD	3 OSU3		3 $\beta \mu$ xxx SUDE3	30 0000 SUDE3	
		E DI	8002			3 β xxxx SUDE3	
		D AD	0003		0003 $\beta \mu$ xxxx		
		S BF	W 0002	6 5222		3 β 0xx μ SUDE3	en W2
6 5222		65	2222				
(6 5222)		R AD	A	0 0	x		
		E DI	W 0002	W 0001		3 β 0xx μ SUDE3	
(W 0001)		3 α	0xx λ	Δ_P	x	décalé	1er décalage
.....					f	(x)	
Sortie		3 β	0xx μ	SUDE3	f (x)		2ème décalage
S UDE3		G NN	1987				
		A RT	1444	1987			
(1987)		T RD	B	N	0 0	f (x)	en B
3 OSU3		30	0000	SUDE3			

- Après assemblage, ce sous-programme a été mis sous forme translatable. Son facteur de translation a été fixé à $D = 1692$.
- Les 3 instructions notées après les réservations sont là à titre indicatif mais ne figurent pas dans le programme PASØ.
- Les instructions de décalage formées ont comme adresse facteur respectivement $0xx\lambda$ et $0xx\mu$ et non 000λ et 000μ . Mais pour les ordres de décalage, seule la position de droite de l'adresse facteur est considérée, à condition toutefois que cette adresse soit valable, ce qui est le cas pour les adresses $0xx\lambda$ et $0xx\mu$.
- La mémoire 1987 est utilisée comme mémoire de travail par le sous-programme auxiliaire de décalage.
- Remarque : Si l'ordre 5P' est indexé, on passe par le sous-programme d'indexage (réservé aux sous-programmes à 3 adresses) avant d'entrer dans le sous-programme de décalage.

b) Sous-programme auxiliaire de décalage en logique extérieure :

P'RA	0000	0029					
P RA	0066	1999					
P'RG	P'1977	1984					
P RG	W 1991	1999					
P'EN	C ØDEC	0030					
P EN	P FDEB	1738					
(D + P')	24	1991	D +10+P'	0..... 0	00 A B	20 $\alpha\lambda\beta\mu$ 1700	en W1
(D+10+P')	69	D +20+P'	D + 30			30 0000 $\Delta_{P'}$	
(D+20+P')	30	0000	$\Delta_{P'}$				

.../...

D+C, 0DEC	T DI	W 0002	0	0	00	A	B	30	0000	Δ P'	en W2
	E DI	6 5222						65	xxxx	W1	
	S BF	6 5222						65	A	W1	en 65222
	D AG	0004		xx	xx	B	0000				
	E DI	2 0SU1						20	0000	SUDE1	
	S BF	1987						20	B	SUDE1	en 1987
	R AD	W 0002	0	0	30	0000	Δ P'				
	E DI	6 9W2						69	W2	xxxx	
	S BI	6 9W2						69	W2	Δ P'	en 69W2
	R AD	W 0001	0	0	20	$\alpha\lambda\beta\mu$	1700				
	T DI	P 0008						20	$\alpha\lambda\beta\mu$	1700	en P8 (analyse)
	D AD	0004						000020	$\alpha\lambda\beta\mu$		
	D AG	0005		2	0	$\alpha\lambda\beta\mu$	000000				
	A AD	3 0SU2						3	$\alpha\lambda\beta\mu$	0SUDE2	30 0000 SUDE2
	R AD	8002	0	0	3	$\alpha\lambda\beta\mu$	0SUDE2		3	$\alpha\lambda\beta\mu$	0SUDE2
	D AD	0003						0003	$\alpha\lambda\beta\mu$	0x	
	S BF	W 0001						3	α	0xx	λ SUDE2
	D AG	0006	0 ...	0xxx	$\beta\mu$	0x0	0				en W1
	S AG	8003	0	0							
	D AD	0001						0	$\beta\mu$	0x0	0
	A AD	3 0SU3						3	$\beta\mu$	0x0	0SUDE3
	E DI	8002									
	D AD	0003						0003	$\beta\mu$	0x0x	
	S BF	W 0002	6 5222					3	0xxx	μ SUDE3	en W2
6 5222	65	2222	W 0001								
(6 5222)	R AD	A	W 0001	0	0		x				
(W 0001)	3	α	0xx	λ	S	UDE2	x	décalé			ler décalage
S UDE2	T RG	P 0005									en P5 et P6
	T RD	P 0006	6 9W2								(analyse)

.../...

6 9W2	69	W 0002	2222				
(6 9W2)	E DI	W 0002	Δ P	x	décalé	33 0xx M SUDE3	
.....				f	(x)		
Sortie	30	0xx M	S UDE3		f (x)		2ème décalage
S UDE3	G NN		1987				
	A RT	1444	1987				
(1987)	T RD	B	S UDE1		f (x)		en B
S UDE1	T DI	P 0007			f (x)		en P7(analyse)
	R AG	8000					
	A NG	P FDEB	D EBUT				
3 OSU2	30	0000	S UDE2				
3 OSU3	30	0000	S UDE3				
2 OSU1	20	0000	S UDE1				

- Après assemblage, ce sous-programme a été mis sous forme translatable. Son facteur de translation a été fixé à D = 1490.
- Les 2ème, 3ème et 4ème paragraphes qui suivent la liste du sous-programme pour l'exécution en langage machine sont valables pour la logique extérieure.
- L'instruction PFDEB est l'instruction n°40 de la logique extérieure, et a comme adresse 1738. Le pseudo-code PEN PFDEB 1738 sert à affecter l'adresse 1738 à PFDEB dans l'assemblage du sous-programme.

.../...

XII - D - 2 - Ordres particuliers au C. D. P. en virgule fixe :

Le C. D. P. en virgule fixe comporte le sous-programme auxiliaire de décalage et 9 entrées différentes dans celui-ci (correspondant chacune à un calcul de fonction). Les codes utilisés sont : $PP' = 50$, $PP' = 51$, ..., $PP' = 58$. Le code 59 reste libre, ce qui permettra d'ajouter au C. D. P. en virgule fixe un sous-programme supplémentaire nécessitant des décalages.

Voici les 9 ordres fonctionnels avec décalages :

50	N iA jB $0 \alpha \lambda \beta \mu$	Racine carrée
51		Sinus
52		Cosinus
53		Arc sinus
54		$\log_{10}$
57		Log_e
55		Translation avec décalage
56		Valeur absolue
58		$e^{(A)}$

Leur emplacement est différent en logique extérieure et en exécution en langage machine.

- En logique extérieure :

$1875 + 50 = 1925$ contient 00 0000 D = 00 0000 1490

..... (D = 1ère instruction du sous-programme auxiliaire de décalage).

$1875 + 5 P' = 1925 + P'$ contient 00 0000 D + P' = 00 0000 149 P'.

$D + 20 + 0 = 1510$ contient 30 0000 $\Delta_0 = 30 0000 1461$

1 = 1511 $\Delta_1 = 1377$

2 = 1512 $\Delta_2 = 1376$

.../...

$D + 20 + 3 = 1513$	contient 30 0000	$\Delta_3 = 30\ 0000$	1186
4 = 1514		$\Delta_4 =$	1316
5 = 1515		$\Delta_5 =$	1809
6 = 1516		$\Delta_6 =$	1261
7 = 1517		$\Delta_7 =$	1317
8 = 1518		$\Delta_8 =$	1263

- En autoprogrammation :

$1875 + 50 = 1925$ contient 00 0000 $D' = 00\ 0000\ 1692$

.....

$1875 = 5\ P' = 1925 + P'$ contient 00 0000 $D' + P' = 00\ 0000\ 1692 + P'$

- En exécution en langage machine :

$D' + 20 + 0 = 1712$	contient 30 0000	$\Delta_0 = 30\ 0000$	1663
1 = 1713		$\Delta_1 =$	1579
2 = 1714		$\Delta_2 =$	1578
3 = 1715		$\Delta_3 =$	1388
4 = 1716		$\Delta_4 =$	1518
5 = 1717		$\Delta_5 =$	1747
6 = 1718		$\Delta_6 =$	1463
7 = 1719		$\Delta_7 =$	1519
8 = 1720		$\Delta_8 =$	1465

.../...

XIII - REMARQUES DIVERSES

XIII - A - CHARGEMENT

Chaque exemplaire du C.D.P. est précédé d'un paquet jaune de chargement. Ce programme de chargement charge non seulement les cartes n mots par carte ($n \leq 7$) ordinaires mais aussi des cartes translatables n mots par carte ($n \leq 5$) de la forme suivante :

Mot 1 : 00 ABCD 000 n NEGATIF

2 : Instruction 1 qui, traduite, entrera en ABCD + Δ

3 : " 2 éventuellement " ... " ABCD + Δ + 1

4 : " 3 " " " ABCD + Δ + 2

5 : " 4 " " " ABCD + Δ + 3

6 : " 5 " " " ABCD + Δ + 4

7 : Indicatif de translation : M_F^1 M_I^1 M_F^2 M_I^2 M_F^5 M_I^5

$\left\{ \begin{array}{l} M_F^x = 8 \text{ si l'adresse F de l'instruction x doit \u00eatre} \\ \text{traduite} \end{array} \right.$

$\left\{ \begin{array}{l} M_F^x = 9 \text{ dans le cas contraire} \end{array} \right.$

$\left\{ \begin{array}{l} M_I^x = 8 \text{ si l'adresse I de l'instruction x doit \u00eatre} \\ \text{traduite} \end{array} \right.$

$\left\{ \begin{array}{l} M_I^x = 9 \text{ dans le cas contraire.} \end{array} \right.$

8 : Indicatif ou blanc.

- Si, pour une carte, $n < 5$, le mot 7 contient les M correspondant

aux n mots et doit \u00eatre obligatoirement compl\u00e9t\u00e9 \u00e0 droite par n'importe

... quoi, par exemple des z\u00e9ros.

.../...

- Les mots négatifs sont également translatés, c'est-à-dire augmentés en valeur absolue.
- Le facteur de translation Δ doit être chargé dans 3 mémoires de la façon suivante :

00	Δ	0000	en 1997
00	0000	Δ	en 1998
00	Δ	Δ	en 1999

Ce chargement peut être fait par une carte n mots par carte non translatable, avec $n = 3$.

- Le programme de chargement utilise les mémoires 1941 à 1999 à part 1950 et 1967. Il est détruit lorsque s'exécute un programme. Inversement, si l'on veut le réutiliser au cours d'un calcul, il détruit les compteurs de boucles, registres d'index et mémoires de travail. Il existe d'ailleurs aussi sous forme translatable (1 instruction par carte).

Dans la liste ci-dessous, les notations suivantes ont été adoptées :

- ABCD est appelé A (pour abréger)
- L'instruction à envoyer en A ou $A + \Delta$ est appelée Ins 0 et non Instruction 1 comme sur la description des cartes translatables .
- L'instruction à envoyer en $A + p$ ou $A + \Delta + p$ est appelée Ins p.. p varie de 0 à $n - 1$.
- $A + \Delta = A'$

.../...

11		T R G	M 1		20 A+p+1 SA	65 L2+p+1 M1	20 A+p+1 SA	en M1
12		T R D	L 0009				65 L2+p+1 M1	en L9
13		S A G	L 0001		p+1-n		20 A + n SA	
14		A N G	S U I T E	L E C T				p + 1 < n ?
15	S U I T E	R A G	L 0010		0 0	0 0		
16		D A G	0002	S D				
17	S D	T R G	L 0010	L 0009			0 0	en L 10
	(L 0009)	R A D	L 2 +p	S B	0 0	Ins p		Cas où transl.
	S B	A N G	Ø U I	N Ø N				Ins p < 0 ?
34	Ø U I	S A G	U N	T E S T	0 0 1	C F I -		
35	N Ø N	A A G	U N	T E S T	0 0 1	C F I		
36	T E S T	E D I	L 0010		0 0 1	C F I +	p p p+1 p+1 M F I F M I	M _F ^p = 8 ?
37		S D O	H U I T	N E U F				M _I ^p = 8 ?
38	H U I T	S D 9	D E L D E	D E L Z E				
40	D E L D E	M U L	D 0003	A D D I T	C F I	00 Δ Δ ±	00 Δ Δ	
41	D E L Z E	M U L	D 0001	A D D I T	C F I	00 Δ 0000 ±	00 Δ 0000	
39	N E U F	S D 9	Z E D E L	M 1				M _I ^p = 8 ?
42	Z E D E L	M U L	D 0002	A D D I T	C F I	00 0000 Δ ±	00 0000 Δ	
43	A D D I T	A A D	8003	M 1	C F I	C F' I' ±	C F I	
	(M 1)	T R D	A +Δ+p	S A			C' F' I' ±	en A + Δ + p
7	(S A)	R A G	M 1		20A'+p SA	0 0		
8	()	A A G	M I L L E		20A'+p+1 SA		00 0001 0000	
9	()	A A D	8001			00 0001 0000		
10	()	A A D	L 0009			65 L2+p+1 SB	65 L2+p SB	
11	()	T R G	M 1				20A'+p+1 SA	en M1
12	()	T R D	L 0009				65L2+p+1 SB	en L 9
13	()	S A G	L 0001		p+1-n			
14	()	A N G	S U I T E	L E C T				p + 1 < n ?
15	(S U I T E)	R A G	L 0010		p p p+1 p+1 M F I F M I	0 0		
16	()	D A G	0002	S D	p+1 p+1 M F M I	00		
17	(S D)	T R G	L 0010	L 0009			M _F ^{p+1} M _I ^{p+1}	en L 10
45	M 1	20	0000	S A				
46	M 2 ..	65	L 0002	S B				
47	M 3	65	L 0002	M 1				
48	M I L L E	00	0001	0000				
49	U N	00	0000	0001				

- Le PEN SC 1994 a comme principale utilité d'obliger SC à être une adresse de la bande 1950 pour que la lecture se fasse bien dans les mémoires 1951 à 1958.
- Dès l'instruction n° 19 se séparent les parties relatives aux cartes translatables et non translatables. Ces parties se rejoignent à CØMMU (n° 25) puis se reséparent à l'instruction n° 31 pour se retrouver en M1.
- Les mémoires L 0009 et L 0010 servent de mémoires de travail.
- Lorsque les 2 branches se rejoignent en CØMMU (n° 25), l'accumulateur droit contient : pour les cartes translatables, 00 Δ 0000, et pour les cartes non translatables, 0.....0 (grâce au fait que l'instruction NTRAN soit un RAG au lieu d'un EDI).
- Si $p + 1 - n \gg 0$, le chargement de la carte est terminé et la carte suivante est lue (test n° 14).
- Les instructions n° 15, 16 et 17 sont inutiles pour les cartes non translatables mais sont conservées pour que les ordres qui les précèdent puissent être communs aux cartes translatables ou non.
- Pour garder une plus grande partie commune aux ordres négatifs et positifs à traduire, on a entré - 1 dans l'accumulateur gauche pour les ordres négatifs (ØUI) et + 1 pour les positifs (NØN).

.../...

- L'introduction des Δ se fait alors, non pas par addition directe, mais par l'intermédiaire d'une multiplication qui affecte la quantité à ajouter à l'instruction du signe convenable (le signe de l'instruction).

XIII - B - FORMATIONS D'ORDRES.

Soit un ordre quelconque à 2 (ou 3) mémoires $PP' N iA jB kC$ (à l'exception d'un ordre de bouclage) dont une des adresses, A par exemple, doit prendre des valeurs diverses, $A_1, A_2, \dots, A_p, \dots$. Supposons par exemple que le programme comporte un ordre de lecture et que les cartes à lire contiennent chacune une valeur A_p . Il est impossible de former l'ordre $PP' N iA_p jB kC$ par substitution d'adresse ou par addition car si l'une de ces opérations est valable en logique extérieure elle ne le sera pas en exécution en langage machine et vice versa. La solution la plus simple est de procéder par indexage.

- Si l'adresse A n'est pas indexée, il suffit d'envoyer A_p dans les positions 1 à 4 d'un registre index inutilisé que nous noterons $n^\circ i$, et d'écrire l'ordre sous la forme : $PP' N i 0000 jB kC$.

- Si l'adresse A est indexée (par un index que nous noterons i'), deux cas peuvent se présenter :

1°) Si l'index i' n'indexe pas d'autre adresse que A , il suffit d'ajouter A_p à la valeur de cet index et d'écrire l'ordre : $PP' N i' 0000 jB kC$. Notons que lorsque l'index i' est remis à zéro et doit encore ultérieurement être utilisé pour former l'ordre $PP' N iA_p jB kC$, il faut y entrer à nouveau A_p .

.../...

2°) Si l'index i' indexe d'autres adresses que A , il faut envoyer $A_p +$ le contenu du registre index n° i' dans un registre index n° i'' inutilisé et écrire l'ordre : PP' N i'' 0000 jB kC. Mais ceci oblige à faire cette opération à chaque modification du registre d'index i' .

XIII- C - INDEXAGE DOUBLE.

Il est très aisé de faire des indexages doubles par addition d'index.

XIII- D - AIGUILLAGE MULTIPLE.

Pour faire un aiguillage multiple c'est-à-dire créer un ordre "Aller à V", V étant variable, il suffit de créer dans une mémoire quelconque l'ordre - 00 0000 V, V ayant la valeur désirée, puis de s'adresser à cette mémoire. Cet ordre doit être négatif pour être valable aussi bien en logique extérieure (- 00 0000 V est considéré comme un ordre en C. D. P.) qu'en exécution en langage machine (- 00 0000 V est considéré comme un ordre en langage machine et son signe est ignoré).

.../...

XIV. CONCLUSION

=====

Le C. D. P. , méthode de programmation intermédiaire entre le FLAIR (logique extérieure où les ordres logiques sont très réduits) et le FORTRAN ou l' A P 3 adaptés aux plus grosses machines , possède en fait maints avantages de chacune des 2 méthodes : comme le FORTRAN il effectue une autoprogrammation et comporte des ordres comparables aux ordres SI, FAIRE, ALLER A,... Mais il a aussi la commodité d'utilisation du FLAIR, et en particulier sa facilité de mise au point.

Le C. D. P. est constitué d'un noyau formé par les opérations en virgule fixe, les ordres logiques et de routine, et les ordres d'entrée dans des sous-programmes. C'est ce noyau qui, de plus, effectue l'indexage et éventuellement l'analyse. L'adjonction aisée de sous-programmes à ce noyau donne une grande souplesse au C. D. P. Le nombre des sous-programmes que l'on peut lui adjoindre simultanément n'est en fait limité que par la capacité du tambour.

On a jusqu'à présent formé trois C. D. P. , d'utilisation fréquente : C. D. P. en virgule fixe, en virgule flottante simple précision, en virgule flottante double précision. A ces C. D. P. peuvent être facilement ajoutés des sous-programmes utiles dans des problèmes particuliers.

Grâce à la souplesse du C. D. P. , on peut aussi, en ajoutant à son noyau des sous-programmes particuliers, former des C. D. P. spécialisés (C. D. P. matriciel, en cours de création ; C. D. P. adapté aux calculs d'impédances et aux calculs en nombres complexes, transposition du FLEC ; etc....).

Le C. D. P. a ainsi de grandes possibilités d'extension. Il sera vraisemblablement aisé de l'adapter aux ordinateurs I. B. M. 650 à registres

.../...

d'index, virgule flottante, bandes et par la suite à d'autres matériels. On pourra même probablement, après avoir mis au point en logique extérieure des programmes sur la 650 I. B. M. , effectuer sur cette même machine une autoprogrammation modifiée qui créera des ordres exécutables sur une autre machine.

Mais dès à présent, grâce à la co-existence de la logique extérieure et de l'autoprogrammation, grâce aussi aux ordres logiques (le bouclage en particulier), le C. D. P. est un outil de travail très commode.

-O-O-O-O-O-O-O-O-O-O-O-

L I S T E S d e s P R O G R A M M E S

I N T E R P R E T A T I F S

=====

1		PRA	1639	1670				
2		PRA	0000	1633				
3		PEN	PPPRI	1750				
4		PEN	ANALY	1700				
5		PEN	001	1764				
6		PEN	PFDEB	1738				
7		PRA	1940	1999				
8		PRG	H1941	1949				
9		PRG	C1968	1976				
10		PRG	P1977	1986				
11		PRG	W1991	1999				
12		PEN	F0000	1846				
13		PRG	F1847	1855				
14		PEN	G0000	1856				
15		PRG	G1857	1865				
16		PEN	T0000	1866				
17		PRG	T1867	1883				
18		PEN	MEMNN	1884				
19		PEN	D0000	1885				
20		PRG	D1886	1939				
21		PER	DEL	1875				
22		PEN	DEBUT	1800				
23	ANALY	RAG	8000	1700	60	8000	1707	
24		ANG	ANA	DEBUT	1707	46	1710	1800
25	ANA	RAG	P0005	1710	60	1981	1685	
26		AAD	P0006	1685	15	1982	1687	
27		DAG	0004	1687	35	4	1697	
28		AAG	EDOKM	8003	1697	10	1701	8003
29	KM	TDI	P0005	1751	24	1981	1684	
30		AAD	EDOKN	8002	1684	15	1737	8002
31	KN	TDI	P0006	1801	24	1982	1735	
32		RAD	P0007	SUANA	1735	65	1983	1787
33	SUANA	EDI	EDOKP	1787	69	1690	1693	
34		SBF	EDOKP	8001	1693	22	1690	8001
35	KP	TDI	P0007	1702	24	1983	1686	
36		AAD	010	1686	15	1689	1743	
37		EDI	EDOKR	1743	69	1696	1699	
38		SBF	EDOKR	8001	1699	22	1696	8001
39	KR	TDI	P0008	PFDEB	1752	24	1984	1738
40	PFDEB	PFO	P0001	DEBUT	1738	71	1977	1800
41	DEBUT	RAD	MEMNN	1800	65	1884	1739	
42		DAG	0004	1739	35	4	1749	
43		TRD	P0001	1749	20	1977	1680	
44		EDI	GG	GI	1680	69	1683	1736
45	GG	RAD	0000	GH	1683	65	1705	
46	GI	SBF	GG	8001	1736	22	1683	8001

47	GH	ANG	UNEME	DEUME	1705	46	1708	1709
48	DEUME	TRD	P0002		1709	20	1978	1681
49		EDI	8003		1681	69	8003	1688
50		SBI	P0005		1688	23	1981	1734
51		SAD	8001		1734	16	8001	1691
52		DAG	0001		1691	35	1	1747
53		AAG	CTE1		1747	10	1802	1757
54		TRG	W0002		1757	21	1992	1695
55		SAG	CTE1		1695	11	1802	1807
56		DAG	0001		1807	35	1	1713
57		SAG	8003		1713	11	8003	1671
58		TDI	FPPRI		1671	24	1750	1703
59		DAG	0003		1703	35	3	1711
60		TRG	MEMNN		1711	21	1884	1837
61		TRD	W0003		1837	20	1993	1746
62		RAD	P0001		1746	65	1977	1731
63		AAD	GJ	8002	1731	15	1784	8002
64	GJ	RAV	0001	GL	1784	67	1	1755
65	GL	TDI	P0003		1755	24	1979	1682
66		EDI	8003		1682	69	8003	1788
67		SBI	P0008		1788	23	1984	1838
68		SAD	8001		1838	16	8001	1745
69		DAG	0001		1745	35	1	1753
70		SAG	8003		1753	11	8003	1761
71		TDI	W0005		1761	24	1995	1698
72		DAG	0004	INNIN	1698	35	4	1759
73	INNIN	TRG	P0006	INCHA	1759	21	1982	1785
74	NINCH	RAG	P0008	COMMU	1803	60	1984	1789
75	COMMU	DAG	0004		1789	35	4	1799
76		AAG	00ANA		1799	10	1704	1809
77		TRG	P0007	W0002	1809	21	1983	1992
78	G0009	SOP	0000	D0051	1865	00		1936
79	F0001	RAD	ODIX0	ADSO1	1847	65	1754	1760
80	F0002	RAD	OONZO	ADSO1	1848	65	1804	1760
81	ADSO1	AAD	P0006		1760	15	1982	1839
82		DAG	0004		1839	35	4	1805
83		AAD	00W3		1805	15	1758	1763
84		TRD	W0002		1763	20	1992	1795
85		RAG	P0005		1795	60	1981	1835
86		DAG	0004		1835	35	4	1845
87		AAD	TG00	KG	1845	15	1748	1706
88	KG	AAG	RGOW2		1706	10	1810	1715
89		AAD	P0007		1715	15	1983	1740
90		TRD	W0003	8003	1740	20	1993	8003
91	F0003	TRD	W0001		1849	20	1991	1694
92		RAG	P0006		1694	60	1982	1790
93		DAG	0004		1790	35	4	1756
94		AAG	ML0W1		1756	10	1811	1765
95		TRG	W0002		1765	21	1992	1796
96		RAD	02KF	KI	1796	65	1806	1712
97	KI	EDI	W0001		1712	69	1991	1744

98		SPI	W0001		1744	23	1991	1794
99		RAG	P0005		1794	60	1981	1786
100		DAG	0004		1786	35		4 1797
101		AAD	TD00	KG	1797	15	1808	1706
102	KF	GNN	ARETM	W0003	1762	44	1815	1993
103	F0005	RAD	P0001		1851	65	1977	1781
104		AAD		8002	1781	15	1834	8002
105		RAD	0002	F0003	1834	65		2 1849
106	F0004	TRD	W0001		1850	20	1991	1844
107		RAD	P0006		1844	65	1982	1840
108		DAG	0004		1840	35		4 1812
109		EDI	KH		1812	69	1716	1719
110		SBF	KH		1719	22	1716	1769
111		EDI	RDAW1		1769	69	1672	1675
112		TDI	W0002		1675	24	1992	1798
113		RAD	02KH	KI	1798	65	1813	1712
114	F0006	RAD	P0001		1852	65	1977	1831
115		AAD		8002	1831	15	1836	8002
116		RAD	0002	F0004	1836	65		2 1850
117	F0007	RAD		F7F8	1853	65	1714	1819
118		46	ANA78		1714	46	1717	1718
119		GNN	EDB	EDA	1718	44	1721	1722
120	EDB	EDI	P0006	EDBA	1721	69	1982	1741
121	EDA	EDI	P0005	EDBA	1722	69	1981	1741
122	EDBA	TDI	MEMNN	ANA78	1741	24	1884	1717
123	F7F8	TRD	W0002		1819	20	1992	1814
124		RAD	P0008		1814	65	1984	1791
125		DAG	0004		1791	35		4 1766
126		AAD	RGOW2	8002	1766	15	1810	8002
127	ANA78	RAG	8000		1717	60	8000	1725
128		ANG	KN	DEBUT	1725	46	1801	1800
129	F0008	RAG	P0006		1854	60	1982	1841
130		DAD	0002		1841	30		2 1816
131		AAD		F7F8	1816	15	1720	1819
132		90	ANA78	EDA	1720	90	1717	1722
133	G0001	RAG	PPPRI	GCOMU	1857	60	1750	1767
134	G0002	RAG	PPPRI	GCOMU	1858	60	1750	1767
135	G0003	RAG	PPPRI	GCOMU	1859	60	1750	1767
136	G0004	RAG	PPPRI	GCOMU	1860	60	1750	1767
137	G0005	RAG	PPPRI	GCOMU	1861	60	1750	1767
138	G0006	RAG	PPPRI	GCOMU	1862	60	1750	1767
139	G0007	RAG	PPPRI	GCOMU	1863	60	1750	1767
140	GCOMU	DAG	0004		1767	35		4 1677
141		AAG	ADDEL	8003	1677	10	1730	8003
142	KJ	EDI	EDP7		1817	69	1770	1673
143		SBI	EDP7		1673	23	1770	1723
144		RAD	P0005		1723	65	1981	1692
145		DAG	0004		1692	35		4 1768
146		AAD	P0006		1768	15	1982	1742
147		AAG	P0007		1742	10	1983	1792
148		AAG	TD00	EDP7	1792	10	1808	1770
149	G0000	RAG	PPPRI		1856	60	1750	1818

150		AAG	CTE2	8003	1818	10	1771	8003
151	F0000	AAD	P0006		1846	15	1982	1842
152		DAG	0004		1842	35	4	1820
153		AAD	P0005		1820	15	1981	1793
154		EDI	KE		1793	69	1821	1674
155		SBI	KE		1674	23	1821	1724
156		RAG	8002		1724	60	8002	1733
157		AAG	RDO1		1733	10	1843	1772
158		AAD	TD00	KE	1772	15	1808	1321
159	CTE1	00	0000	G0000	1802	00		1856
160	010	00	0001	0000	1689	00	1	00
161	0080			80	1822	00		80
162	00ANA	00	0000	ANALY	1704	00		1700
163	001			1	1764	00		1
164	ODIXO	00	0010	0000	1754	00	10	0
165	00W3	00	0000	W0003	1758	00		1993
166	TG00	21	00		1748	21		
167	RGOW2	60	0000	W0002	1810	60		1992
168	MLOW1	19	0000	W0001	1811	19		1991
169	02KF	00	0002	KF	1806	00	2	1762
170	002MI	00	0000	4000	1773	00		4000
171	TD00	20	0		1808	20		
172	ARETM	01	1333	W0003	1815	01	1333	1993
173	KH	64	0000	W0003	1716	64		1993
174	RDAW1	65	8003	W0001	1672	65	8003	1991
175	ADDEL	15	DEL	KJ	1730	15	1875	1817
176	EDP7	11	8003	0000	1770	11	8003	00
177	CTE2	00	0000	F0000	1771	00		1846
178	KE	15	P0007	0000	1821	15	1983	000
179	RDO1	65	0000	0001	1843	65		1
180	EDOKM	69	0000	KM	1701	69		1751
181	EDOKN	69	0000	KN	1737	69		1801
182	EDOKP	69	0000	KP	1690	69		1702
183	02KH	00	0002	KH	1813	00	2	1716
184	00NZO	00	0011	0000	1804	00	11	00
185	EDOKR	69	0000	KR	1696	69		1752
186	UNEME	TRD	F0002		1708	20	1978	1732
187		RAG	8000		1732	60	8000	1823
188		ANG		UNES	1823	46	1676	1727
189		PFO	P0001	UNES	1676	71	1977	1727
190	UNES	RAV	P0002		1727	67	1978	1783
191		EDI	8003		1783	69	8003	1774
192		SBI	MEMNN		1774	23	1884	1824
193		DAG	0002		1824	35	2	1782
194		AAG	CTE4	8003	1782	10	1775	3003
195	T0005	DAD	0006	8002	1871	30	6	3002
196	T0007	EDI	LCDEB	DADSB	1873	69	1726	1679

197	DADSB	DAD	0002	SBFW1	1679	30	2	1825
198	SBFW1	SBF	W0001	8001	1825	22	1991	8001
199	T0001	FDI	ARDEB	DADSB	1867	69	1776	1679
200	T0009	SAG	8003		1875	11	8003	1833
201		DAD	0006		1833	30	6	1826
202		TRD	MTMNN	DEBUT	1826	20	1884	1800
203	CTE4	00	0000	T0000	1775	00		1866
204	LCDEB	70	0000	DEBUT	1726	70		1800
205	ARDEB	01	0000	DEBUT	1776	01		1800
206	T0002	RAG		8003	1868	60	1777	8003
207		TRD	1976	INIT1	1777	20	1976	1729
208	INIT1	SAG	INIT2		1729	11	1832	1827
209		GNN		DEBUT	1827	44	1678	1800
210		AAG		8003	1678	10	1728	8003
211		TRD	1939	INIT1	1728	20	1939	1729
212	INIT2	TRD	1940	INIT1	1832	20	1940	1729
213	T0006	EDI	PFDEB	DADSB	1872	69	1738	1679
214	T0003	SAG	8003		1869	11	8003	1778
215		EDI		DADSB	1778	69	1828	1679
216		TRG	0000	DEBUT	1828	21		1800
217	T0000	SOP	0000	DEBUT	1866	00		1800
218	T0004	DAD	0002		1870	30	2	1779
219		EDI	RGOW2		1779	69	1810	1829
220		SBF	W0001		1829	22	1991	1780
221		EDI	ARDEB		1780	69	1776	1830
222		SBF	W0002	W0001	1830	22	1992	1991

INDEX LOGIQUE EXT

224		PBD	1 639	1 670			
225		PRA	1 671	1 999			
226	INCHA	TRD	W0007		1 785	20	1 997 1 650
227		RAG	P P P R I		1 650	60	1 750 1 655
228		SAG	0080		1 655	11	1 822 1 639
229		ANG	INF	W0002	1 639	46	1 642 1 992
230	INF	RAD	W0003		1 642	65	1 993 1 647
231		DAD	0005		1 647	30	5 1 659
232		AAD	RDCKA	8002	1 659	15	1 662 8002
233	KA	AAG	8001		1 651	10	8001 1 660
234		AAG	P0005		1 660	10	1 981 1 640
235		TRG	P0005		1 640	21	1 981 1 641
236		DAG	0003		1 641	35	3 1 649
237		TRD	P0004		1 649	20	1 980 1 643
238		RAD	W0005		1 643	65	1 995 1 652
239		DAG	0004		1 652	35	4 1 663
240		AAD	RDCKB	8002	1 663	15	1 666 8002
241	KB	AAG	8001		1 653	10	8001 1 661
242		AAG	P0006		1 661	10	1 982 1 644
243		TRG	P0006		1 644	21	1 982 1 645
244		AAD	P0004		1 645	15	1 980 1 646
245		DAG	0003		1 646	35	3 1 656
246		TRD	P0004		1 656	20	1 980 1 648
247		RAD	W0007		1 648	65	1 997 1 654
248		DAD	0005		1 654	30	5 1 667
249		AAD	RDCKC	8002	1 667	15	1 670 8002
250	KC	AAG	8001		1 657	10	8001 1 665
251		AAG	P0008		1 665	10	1 984 1 658
252		AAD	P0004		1 658	15	1 980 1 664
253		SAD	8002		1 664	16	8002 1 668
254		TDI	P0004		1 668	24	1 980 1 669
255		TRG	P0008	COMMU	1 669	21	1 984 1 789
256	RDCKA	RAD	C0000	KA	1 662	65	1 967 1 651
257	RDCKB	RAD	C0000	KB	1 666	65	1 967 1 653
258	RDCKC	RAD	C0000	KC	1 670	65	1 967 1 657

BOUCLAGE LOGIQUE EXT

259		PRA	1500	1999			
260		PBD	0001	0042			
261		PEN	BOU7	0012			
262		PEN	BOU8	0013			
263	0000	RAD	W0003		65	1993	0001
264		DAD	0005		1	30	5 0014
265		AAD	OH00		14	15	17 0021
266		EDI	FOU2		21	69	24 0027
267		SBF	FOU2		27	22	24 0028
268		EDI	FOU4		28	69	31 0034
269		SBF	FOU4		34	22	31 0035
270		EDI	FOU9		35	69	38 0041
271		SBF	FOU9		41	22	38 0042
272		RAD	PPPR1		42	65	1750 0005
273		SAD	0090		5	16	8 0015
274		ANN	BOU1		18	45	18 0019
275		AAD	OCENO	BOU1	0019	15	22 0018
276	BOU1	AAD	OC00		18	15	23 0029
277		DAG	0004		29	35	4 0039
278		EDI	FOU5		39	69	43 0002
279		SBF	FOU5		2	22	43 0003
280		EDI	FOU7		3	69	12 0016
281		SBF	FOU7		15	22	12 0020
282		EDI	FOU11		20	69	25 0030
283		SBF	FOU11		30	22	25 0032
284		AAG	BOU6		32	10	36 0004
285		TRG	W0006	BOU2	0004	21	1996 0024
286	BOU2	RAG	0000	BOU13	0024	60	6
287	BOU3	SAG	P0005		7	11	1981 0037
288		ANG	BOU5	BOU11	0037	46	43 0025
289	BOU13	AAG	001	BOU4	0006	10	1764 0031
290	BOU4	TRG	00		31	21	9
291		TDI	P0004	BOU3	0009	24	1980 0007
292	BOU5	RAG	0000	W0006	0043	60	1996
293	BOU6	AAG	P0008	BOU7	0036	10	1984 0012
294	BOU7	TRG	00		12	21	10
295		TDI	P0005	BOU6	0010	24	1981 0013
296	BOU8	EDI	P0000		13	69	1982 0040
297		TDI	MEMNN	ANABO	0040	24	1984 0011
298	BOU11	TRD	0000	BOU9	0025	20	38
299	BOU9	TDI	0000	ANABO	0038	24	11
300	ANABO	RAG	8000		11	60	8000 0026
301		ANG	PFDEB	DEBUT	0026	40	1738 1800
302	OH00	00	H00		17	00	1940 0
303	0090			90	0006	00	90
304	OCENO	00	0100		22	00	100
305	OC00	0		00000	0023	00	1967

AUTO PROGRAMMATION

1		PRA	0000	1351				
2		PRA	1940	1999				
3		PRG	H1941	1949				
4		PRG	C1968	1976				
5		PRG	P1977	1986				
6		PRG	W1991	1999				
7		PRG	X1838	1846				
8		PRG	F1847	1855				
9		PEN	G0000	1850				
10		PRG	G1657	1865				
11		PEN	T0000	1866				
12		PRG	T1867	1881				
13		PEN	CTRL	1882				
14		PEN	LMAX	1883				
15		PEN	MEMNN	1884				
16		PEN	D0000	1885				
17		PRG	D1886	1939				
18		PEN	DEBUT	1800				
19		PER	K	1930				
20		PER	KPLU2	1932				
21		PER	ARETM	1936				
22		PER	Z	1939				
23	DEBUT	RAD	MEMNN	1800	65	1884	1389	
24		DAG	0004	1389	35	4	1399	
25		TRD	MEMN	1399	20	1403	1356	
26	DEBU2	TRG	P0003	1356	21	1979	1382	
27		TDI	P0004	1382	24	1980	1383	
28		TDI	P0005	1383	24	1981	1384	
29		TDI	P0006	1384	24	1982	1385	
30		TDI	P0007	1385	24	1983	1386	
31		TDI	P0008	1386	24	1984	1387	
32		EDI	FF	1387	69	1390	1393	
33	FF	RAD	0000	1390	65		1355	
34	FG	S5F	FF	1393	22	1390	8001	
35	FH	ANG	UNEME	1355	46	1358	1359	
36	DEUME	EDI	8003	1359	69	8003	1366	
37		SBI	00001	1366	23	1968	1371	
38		SAD	8001	1371	16	8001	1379	
39		DAG	0001	1379	35	1	1435	
40		AAG	CTE1	1435	10	1388	1443	
41		TRG	W0002	1443	21	1992	1395	
42		SAG	CTE1	1395	11	1388	1493	
43		DAG	0001	1493	35	1	1449	
44		SAG	8003	1449	11	8003	1357	
45		TDI	PPPRI	1357	24	1360	1363	
46		DAG	0003	1363	05	3	1421	
47		TRG	MEMNN	1421	21	1884	1437	
48		TRD	00004	1437	20	1971	1374	

49		RAD	MEMN						
50		AAD	002		1 374	65	1 403	1 407	
51		TRD	P0001		1 407	15	1 410	1 365	
52		AAD	010		1 365	20	1 977	1 380	
53		EDI	FJ	FK	1 380	15	1 433	1 487	
54	FJ	RAV	0000	FL	1 487	69	1 440	1 543	
55	FK	SBF	FJ	8001	1 440	67		1 405	
56	FL	EDI	8003		1 543	22	1 440	8001	
57		SBI	C0003		1 405	69	8003	1 362	
58		SAD	8001		1 362	23	1 970	1 373	
59		DAD	0001		1 373	16	8001	1 381	
60		EDI	8003		1 381	30	1	1 537	
61		SBF	C0002		1 537	69	8003	1 394	
62		SAD	8001		1 394	22	1 969	1 372	
63		DAD	0003		1 372	16	8001	1 429	
64		EDI	8003		1 429	30	3	1 587	
65		SBI	C0005		1 587	69	8003	1 444	
66		DAD	0001		1 444	23	1 972	1 375	
67		SBF	C0005		1 375	30	1	1 431	
68		RAG	C0003		1 431	22	1 972	1 425	
69		DAG	0004		1 425	60	1 970	1 475	
70		AAG	MEMNN		1 475	35	4	1 485	
71		TRG	C0006	W0002	1 485	10	1 884	1 439	
72	PERFO	PFO	P0001		1 439	21	1 973	1 992	
73		AAG	CTRL		1 400	71	1 977	1 377	
74		DAG	0004		1 377	10	1 882	1 637	
75		TRG	P0001		1 637	35	4	1 397	
76		AAG	8002		1 397	21	1 977	1 430	
77		DAD	0004		1 430	10	8002	1 489	
78		TRG	CTRL		1 489	30	4	1 499	
79		SAG	LMAX		1 499	21	1 882	1 535	
80		SAG	002	SUITE	1 535	11	1 883	1 687	
81	SUITE	EDI	W0001		1 687	11	1 410	1 415	
82		TDI	P0002		1 415	69	1 991	1 494	
83		EDI	W0002		1 494	24	1 978	1 481	
84		TDI	P0003		1 481	69	1 992	1 445	
85		EDI	W0003		1 445	24	1 979	1 432	
86		TDI	P0004		1 432	69	1 993	1 396	
87		ANG	FINPF	ARETL	1 396	24	1 980	1 483	
88	FINPF	PFO	P0001	DEBUT	1 483	46	1 436	1 737	
108	G0009	RAG	CTRL		1 436	71	1 977	1 800	
109		DAG	0004		1 865	60	1 882	1 438	
110		AAG	8001		1 438	35	4	1 599	
111		AAG	RG01		1 599	10	8001	1 507	
112		TRG	P0002		1 507	10	1 510	1 565	
113		AAD	MEMN		1 565	21	1 978	1 581	
114		AAD	D0051		1 581	15	1 403	1 557	
115		AAD	ED10		1 557	15	1 936	1 491	
116		TRD	W0002		1 491	15	1 644	1 649	
					1 649	20	1 992	1 545	

117		DAD	0006		1545	30	6	1409
118		SHF	W0001		1409	22	1991	1694
119		EDI	C0006		1694	69	1973	1426
120		TDI	W0003		1426	24	1993	1496
121		RAG	PPPRI		1496	60	1360	1615
122		DAD	0002		1615	30	2	1571
123		AAD	C0002		1571	15	1969	1473
124		DAD	0003		1473	30	3	1631
125		AAD	C0004		1631	15	1971	1575
126		DAD	0001		1575	30	1	1681
127		AAD	MCTRO		1681	15	1486	1541
128		TRD	P0005		1541	20	1981	1434
129		EDI	C0001		1434	69	1968	1621
130		TDI	P0003	VAPF4	1621	24	1979	1482
131	G0000	RAD	PPPRI		1856	65	1360	1665
132		ANN	VFX	SP2A	1665	45	1368	1369
133	VFX	AAG	C0005		1368	10	1972	1427
134		AAG	C0004		1427	10	1971	1625
135		GNN	IND1	NIND1	1625	44	1479	1480
136	IND1	AAD	CTE2		1479	15	1532	1488
137		TRD	W0004		1488	20	1994	1497
138		TRG	W0001		1497	21	1991	1744
139		RAD	CTRL		1744	65	1882	1538
140		DAG	0004		1538	35	4	1699
141		AAD	D0052		1699	15	1937	1591
142		AAG	8001		1591	10	3001	1749
143		AAG	MEMN		1749	10	1403	1607
144		AAG	RD10		1607	10	1560	1715
145		TRG	P0002		1715	21	1978	1731
146		AAD	6013		1731	15	1484	1539
147		TRD	P0003		1539	20	1979	1582
148		RAD	C0002	W0004	1582	65	1969	1994
149	F0001	AAD	AGOW2	ADSO1	1847	15	1450	1455
150	F0002	AAD	SGOW2	ADSO1	1848	15	1401	1455
151	ADSO1	TRD	W0003		1455	20	1993	1546
152		RAD	C0001		1546	65	1968	1523
153		DAG	0004		1523	35	4	1583
154		AAD	RGOW1		1583	15	1536	1641
155		TRD	P0005		1641	20	1981	1534
156		AAG	TG00	ASMU1	1534	10	1588	1593
157	ASMU1	AAG	C0006		1593	10	1973	1477
158		TRG	W0002	VAPF4	1477	21	1992	1482
159	F0003	AAD	MULOZ		1849	15	1402	1657
160		TRD	W0003	REUN1	1657	20	1993	1596
161	REUN1	RAG	C0001		1596	60	1968	1573
162		DAG	0004		1573	35	4	1633
163		AAG	RGOW1	MUDI1	1633	10	1536	1691
164	MUDI1	TRG	P0005		1691	21	1981	1584
165		RAG	TD00	ASMU1	1584	60	1638	1593
166	F0005	AAD	MULOZ		1851	15	1402	1707

167		TRD	W0003			1707	20	1993	1646
168		RAD	P0001			1646	65	1977	1781
169		AAD	001			1781	15	1634	1589
170		TRD	P0001			1589	20	1977	1530
171		DAD	0004			1530	30		4
172		AAD	002			1741	15	1410	1765
173		EDI	W0003			1765	69	1993	1696
174		SBI	W0003			1696	23	1993	1746
175		DAG	0004			1746	35		4
176		AAD	EDOGA	8002		1757	15	1610	8002
177	GA	TDI	P0004	REUN1		1500	24	1980	1596
178	F0004	AAD	DROW2			1850	15	1453	1807
179		TRD	W0003			1807	20	1993	1796
180		RAG	C0001			1796	60	1968	1623
181		DAG	0004			1623	35		4
182		AAG	00W1	REUN2		1683	10	1586	1791
183	REUN2	AAG	RD00	MUDI1		1791	10	1794	1691
184	F0006	AAD	DROW2			1852	15	1453	1408
185		TRD	W0003			1408	20	1993	1547
186		RAD	P0001			1547	65	1977	1831
187		AAD	001			1831	15	1634	1639
188		TRD	P0001			1639	20	1977	1580
189		AAD	020			1580	15	1733	1688
190		EDI	ED0GB			1688	69	1392	1595
191		SBF	ED0GB	8001		1595	22	1392	8001
192	GP	TDI	P0004			1550	24	1980	1783
193		DAD	0004			1783	30		4
194		DAG	0006			1643	35		6
195		AAG	C0001			1458	10	1968	1673
196		DAG	0004	REUN2		1673	35		4
197	F0007	AAD	C0001			1853	15	1968	1723
198		AAG	4400			1723	10	1476	1632
199		TRD	P0005			1632	20	1981	1684
200		RAD	MEMNN			1684	65	1884	1689
201		AAG	C0003			1689	10	1970	1675
202		DAG	0004			1675	35		4
203		AAG	RG00			1585	10	1738	1693
204		AAD	CTRL			1693	15	1882	1788
205		AAD	4603			1788	15	1442	1597
206		EDI		F7F8		1597	69	1600	1503
207		TRD	W0001	VAPF4		1600	20	1991	1482
208	4400	44	00			1476	44		
209	4603	46	0000	0003		1442	46		3
210	F0008	AAD	MEMNN			1854	15	1884	1739
211		AAG	C0003			1739	10	1970	1725
212		DAG	0004			1725	35		4
213		AAG	69			1635	10	1594	1799
214		AAD	C0001			1799	15	1968	1773
215		AAD	900			1773	15	1526	1682

216		EDI		F7F5	1682	69	1685	1503
217		TRD	W0001	VAPF3	1685	20	1991	1645
218	900	90	0		1526	90		
219	F7F8	TDI	W0005		1503	24	1995	1398
220		TRD	W0003		1398	20	1993	1647
221		AAG	CTRL		1647	10	1882	1789
222		AAG	002		1789	10	1410	1815
223		TRG	W0002		1815	21	1992	1695
224		RAD	MEMN		1695	65	1403	1508
225		DAG	0002		1508	35	2	1416
226		AAG	C0005		1416	10	1972	1527
227		AAD	CTRL		1527	15	1882	1490
228		EDI	DAG4		1490	69	1743	1697
229		SBI	P0003	DAG4	1697	23	1979	1743
230	DAG4	DAG	0004		1743	35	4	1553
231		AAG	RGC01		1553	10	1406	1361
232		TRG	P0002		1361	21	1978	1732
233		AAD		W0005	1732	15	1735	1995
234		10	0001	8003	1735	10	1	8003
235	RGC01	60	C0000	0001	1406	60	1967	0001
236	NIND1	AAD	CTE3		1480	15	1833	1540
237		TRD	W0004		1540	20	1994	1747
238		RAG	C0001		1747	60	1968	1823
239		DAD	0002		1823	30	2	1529
240		AAD	MEMN		1529	15	1403	1558
241		DAG	0006		1558	35	6	1424
242		AAG	001		1424	10	1634	1590
243		AAD	C0002	W0004	1590	15	1969	1994
244	X0001	AAD	AG00	ADS02	1838	15	1492	1797
245	X0002	AAD	SG00	ADS02	1839	15	1542	1797
246	ADS02	AAD	CTRL		1797	15	1882	1640
247		TRD	P0003		1640	20	1979	1782
248		AAG	RG00		1782	10	1738	1793
249		TRG	P0002		1793	21	1978	1832
250		RAG	TG00	ASDIV	1832	60	1588	1745
251	ASDIV	AAG	C0006		1745	10	1973	1577
252		TRG	W0001	VAPF1	1577	21	1991	1795
253	X0003	AAG	RG00		1840	10	1738	1448
254		TRG	P0002		1448	21	1978	1734
255		RAG	8002		1734	60	8002	1498
256		AAG	MULOZ		1498	10	1402	1608
257		AAD	CTRL		1608	15	1882	1690
258		AAD	001		1690	15	1634	1740
259		EDI	8003	REUN3	1740	69	8003	1548
260	REUN3	SBI	P0003		1548	23	1979	1784
261		RAG	CTRL		1784	60	1882	1790
262		AAG	GNARM		1790	10	1598	1603
263		TRG	W0002		1603	21	1992	1648
264		AAD	C0006		1648	15	1973	1627

265		AAD	TD00		1627	15	1638	1638
266		TRD	W0001	VAPF2	1698	20	1991	1748
267	X0005	AAG	RG00		1842	10	1738	1798
268		TRG	P0002		1798	21	1978	1834
269		RAG	8002		1834	60	8002	1650
270		AAD	P0001		1650	15	1977	1785
271		AAD	001		1785	15	1634	1592
272		TRD	P0001		1592	20	1977	1630
273		AAD	020		1630	15	1733	1642
274		AAD	CTRL		1642	15	1882	1692
275		SAD	002		1692	16	1410	1466
276		EDI	ED2GC		1466	69	1419	1422
277		SBF	ED2GC	8001	1422	22	1419	8001
278	GC	SBI	P0004		1700	23	1980	1835
279		AAG	MULOZ		1835	10	1402	1658
280		EDI	8003		1658	69	8003	1364
281		DAD	0004	REUN3	1364	30	4	1548
282	X0004	AAD	DR00		1341	15	1750	1505
283		AAD	CTRL		1505	15	1882	1742
284		TRD	P0003		1742	20	1979	1636
285		AAG	RD00		1636	10	1794	1451
286		TRG	P0002	REUN4	1451	21	1978	1686
287	REUN4	RAG	TD00	ASDIV	1686	60	1638	1745
288	X0006	AAD	DR00		1843	15	1750	1555
289		AAD	CTRL		1555	15	1882	1792
290		SAD	8002		1792	16	8002	1501
291		TDI	P0003		1501	24	1979	1736
292		AAD	P0001		1736	15	1977	1786
293		AAD	001		1786	15	1634	1551
294		TRD	P0001		1551	20	1977	1680
295		AAD	020		1680	15	1733	1601
296		EDI	EDOGD		1601	69	1354	1708
297		SBF	EDOGD		1708	22	1354	1758
298		PAD	8003	EDOGD	1758	65	8003	1334
299	GD	SBI	P0004		1651	23	1980	1836
300		AAD	RD01		1836	15	1701	1605
301		TRD	P0002	REUN4	1605	20	1978	1636
302	X0007	AAD	C0001		1844	15	1968	1474
303		AAD	4400		1474	15	1476	1751
304		TRD	W0001		1751	20	1991	1601
305		RAD	MEMN		1801	65	1403	1808
306		DAG	0002		1808	35	2	1516
307		AAG	C0003		1516	10	1970	1775
308		AAD	MEMNN		1775	15	1884	1452
309		DAG	0004		1452	35	4	1413
310		AAG	RG01		1413	10	1510	1566
311		AAD	CTRL		1566	15	1882	1502
312		AAD	4600		1502	15	1655	1459
313		TRD	P0003	TGP21	1459	20	1979	1552

314	4600	46	00		1655	46		
315	X0008	AAD	MEMNN		1845	15	1884	1602
316		DAG	0004		1602	35	4	1463
317		AAD	C0001		1463	15	1960	1524
318		AAD	900		1524	15	1526	1652
319		TRD	P0003		1652	20	1979	1702
320		RAD	MEMN		1702	65	1403	1509
321		DAG	0002		1509	35	2	1616
322		AAD	C0003		1616	15	1970	1825
323		DAD	0006		1825	30	6	1752
324		AAD		VAPF0	1752	15	1705	1559
325		69	0000	0001	1705	69		1
326	SP2A	RAG	C0001		1369	60	1968	1574
327		AAG	C0002		1574	10	1969	1624
328		AAG	RD01		1624	10	1701	1755
329		TRG	W0001		1755	21	1991	1802
330		RAD	C0006		1802	65	1973	1677
331		AAD	TD00		1677	15	1638	1653
332		AAG	C0005		1653	10	1972	1727
333		GNN	IND2A	NIN2A	1727	44	1703	1753
334	IND2A	EDI	W0001		1703	69	1991	1803
335		TDI	W0002		1803	24	1992	1404
336		TRG	W0001		1404	21	1991	1454
337		TRD	W0003		1454	20	1993	1504
338		RAD	CTRL		1504	65	1882	1554
339		DAG	0004		1554	35	4	1666
340		AAG	MEMN		1666	10	1403	1609
341		AAG	RG1AG		1609	10	1460	1716
342		AAD	D0052		1716	15	1937	1604
343		AAD	MTMX		1604	15	1659	1513
344		EDI	8003		1513	69	8003	1370
345		SBI	P0002		1370	23	1978	1654
346		AAD	RG08		1654	15	1709	1563
347		TRD	P0003	VAPF3	1563	20	1979	1645
348	NIN2A	TRD	W0002		1753	20	1992	1704
349		RAD	MEMN		1704	65	1403	1759
350		DAG	0002		1759	35	2	1766
351		AAG	CTRL		1766	10	1882	1754
352		AAD	8001		1754	15	8001	1411
353		DAG	0004		1411	35	4	1671
354		AAG	RG01		1671	10	1510	1816
355		TRG	P0002		1816	21	1978	1804
356		AAD	C0001		1804	15	1968	1674
357		AAD	AD10	TDP32	1674	15	1777	1805
358	TDP32	TRD	P0003	VAPF2	1805	20	1979	1748
359	G0001	RAD	C0002	SUITG	1857	65	1969	1724
360	G0002	RAD	C0002	SUITG	1859	65	1969	1724
361	G0003	RAD	C0002	SUITG	1859	65	1969	1724
362	G0004	RAD	C0002	SUITG	1860	65	1969	1724

363	G00005	RAD	C00002	SUITG	1861	65	1969	1724
364	SUITG	DAG	0002		1724	35	2	1456
365		AAG	C00001		1456	10	1968	1774
366		DAG	0004		1774	35	4	1506
367		TRG	W0001		1506	21	1991	1556
368		RAD	PPPRI		1556	65	1360	1367
369		DAG	0004		1367	35	4	1827
370		AAD	EDDEL		1827	15	1730	1606
371		SAD	ODIXO	8002	1606	16	1809	8002
372	HA	TDI	W0004		1656	24	1994	1706
373		RAD	C0006		1706	65	1973	1378
374		AAD	TD00		1378	15	1638	1756
375		AAG	C0004		1756	10	1971	1576
376		AAG	C0005		1576	10	1972	1428
377		GNN	IND3A	NIN3A	1428	44	1806	1660
378	IND3A	EDI	W0001		1806	69	1991	1710
379		TDI	W0002		1710	24	1992	1760
380		TRG	W0001		1760	21	1991	1810
381		TRD	W0003		1810	20	1993	1461
382		RAG	CTRL		1461	60	1882	1511
383		DAG	0004		1511	35	4	1721
384		AAG	W0004		1721	10	1994	1501
385		AAG	RG00		1501	10	1738	1611
386		AAD	D0052		1611	15	1937	1661
387		AAD	MEMY		1661	15	1414	1469
388		AAD	MEMN		1469	15	1403	1711
389		AAD	RD10		1711	15	1560	1417
390		TRG	P0002	TGP33	1417	20	1978	1549
391	TGP33	TRG	P0003	VAPF3	1549	21	1979	1645
392	NIN3A	TRD	W0002		1660	20	1992	1701
393		RAD	MEMN		1701	65	1403	1811
394		DAG	0002		1811	35	2	1467
395		AAG	CTRL		1467	10	1882	1412
396		AAD	8001		1412	15	8001	1519
397		DAG	0004		1519	35	4	1579
398		AAG	RD01		1579	10	1701	1402
399		TRG	P0002		1402	21	1978	1512
400		AAD	W0004		1512	15	1994	1562
401		AAD	ED10	TDP32	1562	15	1644	1805
402	CTE1	00	0000	00000	1388	00		1856
403	002			2	1410	00		2
404	010	00	0001	0000	1433	00	1	00
405	RG1AG	60	0001	8003	1460	60	1	8003
406	MCTRO	00	1877	0000	1486	00	1877	00
407	69	69	00		1594	69		
408	PERF1	00	0100	0001	1612	00	100	0001
409	PERF2	00	0200	0002	1662	00	200	0002
410	PERF3	00	0300	0003	1712	00	300	0003

411	PERF4	00	0400	0004	1762	00	400	0004
412	PERF5	00	0500	0005	1812	00	500	0005
413	CTE2	00	0000	F0000	1532	00		1846
414	CTE3	00	0000	X0000	1833	00		1837
415	RD10	65	0001	0000	1560	65	1	00
416	6013	60	0000	0013	1434	60		13
417	AGOW2	10	0000	W0002	1450	10		1992
418	SGOW2	11	0000	W0002	1401	11		1992
419	TG00	21	00		1560	21		
420	RGOW1	60	0000	W0001	1536	60		1901
421	MULOZ	19	0000	Z	1402	19		1959
422	001			1	1634	00		1
423	EDOGA	69	0000	GA	1610	69		1500
424	TDOO	20	0		1638	20		
425	DROW2	64	0000	W0002	1453	64		1992
426	00W1	00	0000	W0001	1566	00		1991
427	020	00	0002	0000	1733	00	2	00
428	EDOGB	69	0000	GB	1392	69		1550
429	RD00	65	00		1794	65		
430	AG00	10	0		1492	10		
431	SG00	11	00		1542	11		
432	RG00	60	0		1738	60		
433	ED2GC	69	0002	GC	1419	69	2	1700
434	GNARM	44	ARETM	0000	1598	44	1936	000
435	DROO	64	00		1750	64		
436	EDOGD	69	0000	GD	1354	69		1651
437	RD01	65	0000	0001	1701	65		1
438	RG08	60	0000	8	1709	60		8
439	MEMX	0		28	1659	00		28
440	MEMY	0		48	1414	00		48
441	VAPF1	RAD	PERF1	PERFO	1795	65	1612	1400
442	VAPF2	RAD	PERF2	PERFO	1748	65	1662	1400
443	VAPF3	RAD	PERF3	PERFO	1645	65	1712	1400
444	VAPF4	RAD	PERF4	PERFO	1482	65	1762	1400
445	VAPF5	RAD	PERF5	PERFO	1613	65	1812	1400
446	RG01	60	000	1	1510	60		1
447	AD10	15	0001	0000	1777	15	1	00
448	EDDEL	69	DO000	HA	1730	69	1885	1656
449	ODIXO	00	0010	0000	1809	00	10	0
450	ED10	69	0001	0000	1644	69	1	00
451	033	00	0003	0003	1663	00	3	0003
452	24W12	24	W0001	0002	1713	24	1991	0002
453	240K	24	0000	K	1763	24		1930
454	24551	24	0055	0001	1813	24	55	0001
455	44K2	44	KPLU2	0000	1464	44	1932	000
456	ARETL	LEC	1951		1737	70	1951	1514
457	ED1	ED1	1951		1514	69	1951	1564

458		TDI	CTRL		1864	24	1862	1614
459		EDI	1882		1614	69	1952	1664
460		TDI	LMAX		1664	24	1883	1714
461		RAD	MEMN	DEBU2	1714	65	1403	1356
462	UNEME	TRD	W0002		1358	20	1992	1764
463		RAD	MEMN		1764	65	1403	1814
464		AAD	001		1814	15	1634	1517
465		TRD	P0001		1517	20	1977	1780
466		RAV	W0002		1780	67	1992	1567
467		EDI	8003		1567	69	8003	1824
468		SBI	MEMNN		1824	23	1884	1617
469		EDI	8003		1617	69	8003	1626
470		SBF	W0001		1626	22	1991	1667
471		DAG	0002		1667	35	2	1676
472		AAG	CTE4	8003	1676	10	1629	5003
473	T0005	RAD	W0001		1871	65	1991	1717
474		DAD	0004		1717	30	4	1478
475		TRD	W0001	UNE53	1478	20	1991	1767
476	UNE53	TRD	P0002		1767	20	1978	1817
477		PFO	P0001		1817	71	1977	1528
478		RAD	W0001		1528	65	1991	1418
479		DAG	0004		1418	35	4	1679
480		AAD	001		1679	15	1634	1468
481		TRD	P0001		1468	20	1977	1830
482		EDI	UNE51		1830	69	1518	1771
483		SBF	UNE51	8001	1771	22	1518	8001
484	UNE51	RAV	00		1518	67		1568
485		EDI	8003		1568	69	8003	1726
486		SBI	W0001		1726	23	1991	1618
487		AAG	8001		1618	10	8001	1776
488		SAG	ENCOD		1776	11	1729	1668
489		GNN		UNE52	1668	44	1821	1472
490		AAG	8001	UNE53	1821	10	8001	1767
491	UNE52	RAG	8002		1472	60	8002	1718
492		AAD	MEMNN		1718	15	1884	1768
493		SAG	8003		1768	11	8003	1826
494		SBI	P0002	FINPF	1826	23	1978	1436
495	T0007	RAD	6300	LECPF	1873	65	1578	1818
496	LECPF	AVD	W0002	VAPFO	1818	17	1992	1559
497	VAPFO	TRD	P0002	FINPF	1559	20	1978	1436
498	T0001	RAV	W0002	VAPFO	1867	67	1992	1559
499	T0009	DAD	0006	VAPFO	1875	30	6	1559
500	CTE4	00	0000	T0000	1829	00		1866
501	ENCOD	00	0000	DEBUT	1729	00		1800
502	6300	63	00		1578	63		
503	T0002	RAG	CTRL		1868	60	1882	1569
504		DAG	0004		1569	35	4	1779
505		AAG	8001		1779	10	8001	1618
506		AAD	8001		1619	15	8001	1628

507		AAG	RG01		1628	10	1510	1669
508		TRG	PO002		1669	21	1578	1719
509		AAG	033		1719	10	1663	1769
510		TRG	W0003		1769	21	1993	1819
511		AAD	24W12		1819	15	1713	1420
512		TRD	W0002		1420	20	1392	1470
513		RAD	D0051		1470	65	1936	1520
514		DAG	0004		1520	35	4	1570
515		AAD	8001		1570	15	8001	1678
516		EDI	240K		1678	69	1753	1620
517		SEF	PC000		1620	22	1982	1670
518		AAD	24551		1670	15	1813	1720
519		TRD	PO005		1720	20	1981	1770
520		RAG	MEMNN		1770	60	1884	1820
521		AAG	44K2		1820	10	1464	1522
522		TRG	W0001	VAPF5	1522	21	1991	1613
523	T0006	RAD	RD00	LECPF	1872	65	1794	1818
524	T0003	SAG	8003		1869	11	8003	1728
525		DAD	0002		1728	30	2	1572
526		AAD	TD00		1572	15	1638	1622
527		TRD	W0001		1622	20	1991	1672
528		RAG	CTRL		1672	60	1882	1722
529		AAG	RG00	TGP21	1722	10	1738	1552
530	TGP21	TRG	PO002	VAPF1	1552	21	1978	1795
531	T0000	RAV	W0002	VAPF0	1866	67	1992	1552
532	T0004	RAV	W0002		1870	67	1992	1772
533		SAD	0300		1772	16	1778	1822
534		TRD	W0001		1822	20	1991	1828
535		AAG	69		1828	10	1594	1829
538		AAG	CTRL		1829	10	1682	1352
539		SAG	8003		1352	11	2003	1353
540		SEF	PO002	VAPF1	1353	22	1978	1795
541	0300	03	00		1778	03		

CHARGEMENT

1		PRA	1995	1995				
		PRA	1967	1967				
		PRA	0000	1940				
3		PRA	1950	1950				
4		PRG	D1997	1999				
5		PRG	L1951	1960				
6		PEN	SC	1994				
7	SA	RAG	M1		1961	60	1964	1969
8		AAG	MILLE		1969	10	1972	1977
9		AAD	8001		1977	15	8001	1985
10		AAD	L0009		1985	15	1959	1963
11		TRG	M1		1963	21	1964	1967
12		TRD	L0009		1967	20	1959	1962
13		SAG	L0001		1962	11	1951	1965
14		ANG	SUITE	LECT	1965	46	1968	1970
15	SUITE	RAG	L0010		1968	60	1960	1966
16		DAG	0002	SD	1965	35	2	1973
17	SD	TRG	L0010	L0009	1973	21	1960	1959
18	SC	RAD	L0001		1994	65	1951	1971
19		ANG	TRAN	NTRAN	1971	46	1974	1975
20	NTRAN	RAG	M3	COMMU	1975	60	1978	1983
21	TRAN	RAD	D0001		1974	65	1997	1976
22		EDI	L0007		1976	69	1957	1979
23		TDI	L0010		1979	24	1960	1980
24		EDI	M2	COMMU	1980	69	1984	1983
25	COMMU	TDI	L0009		1983	24	1959	1981
26		AVD	L0001		1981	17	1951	1982
27		EDI	M1		1982	69	1964	1986
28		SRF	M1		1985	22	1964	1987
29		DAG	0004		1987	35	4	1947
30		AAD	8001		1947	15	8001	1988
31		SRF	L0001	L0009	1984	22	1951	1959
33	SB	ANG	OUI	NON	1989	46	1942	1943
34	OUI	SAG	UN	TEST	1942	11	1945	1949
35	NON	AAG	UN	TEST	1943	10	1945	1949
36	TEST	EDI	L0010		1949	69	1960	1990
37		SD0	HUIT	NEUF	1990	90	1944	1946
38	HUIT	SD9	DELDE	DELZE	1944	99	1948	1991
39	NEUF	SD9	ZEDEL	M1	1946	99	1992	1964
40	DELDE	MUL	D0003	ADDIT	1948	19	1999	1993
41	DELZE	MUL	D0001	ADDIT	1991	19	1997	1993
42	ZEDEL	MUL	D0002	ADDIT	1992	19	1998	1993
43	ADDIT	AAD	8003	M1	1993	15	8003	1964
44	LECT	LEC	SC	8000	1970	70	1994	8000
45	M1	TRD	0000	SA	1964	20		1961
46	M2	RAD	L0002	SB	1984	65	1952	1989
47	M3	RAD	L0002	M1	1978	65	1952	1964
48	MILLE	OO	0001	0000	1972	00	1	00
49	UN			1	1945	00		1

Vu et Approuvé

Nancy, le

Le Doyen,

E. URION.

Vu et Permis d'Imprimer

Nancy, le

Le Recteur :

Président du Conseil de

l'Université.

P. IMBS.

